# Learning a Pile Loading Controller from Demonstrations

Wenyan Yang*,[1], Nataliya Strokina*,[1], Nikolay Serbenyuk*,[2], Reza Ghabcheloo[2] and Joni Kämäräinen[1]

***Abstract*— This work introduces a learning-based pile loading controller for autonomous robotic wheel loaders. Controller parameters are learnt from a small number of demonstrations for which low level sensor (boom angle, bucket angle and hydrostatic driving pressure), egocentric video frames and control signals are recorded. Application specific deep visual features are learnt from demonstrations using a Siamese network architecture and a combination of cross-entropy and contrastive loss. The controller is based on a Random Forest (RF) regressor that provides robustness against changes in field conditions (loading distance, soil type, weather and illumination). The controller is deployed to a real autonomous robotic wheel loader and it outperforms prior art with a clear margin.**

## I. INTRODUCTION

Heavy-duty mobile (HDM) working machines are the workhorse of the earth moving industry. Autonomy and operator assistant functions for these machines will have tremendous direct impact on productivity and safety in construction industry. Among the functions these machines do, pile loading is one of the most demanding due to modeling challenges and variation in the surface material. Although operators learn to operate these complex machines, there is still no reliable automation solution in industry or academia for this task. Surge of machine learning techniques and our understanding of their potentials and limits motivated us to address pile loading problem employing advances in machine learning and benefiting from operator demonstrations. We show how operator's skills can be transferred to the machine by a relatively low number of demonstrations.

Majority of the existing approaches to automation of pile loading are blind. They rely on low level sensory data of the machines whereas a professional operator always observes the pile condition. Earlier works solve the problem using the classical optimal control tools, for example, by optimizing the power transmitted from the machine to soil [1], [2]. In a recent work [3] a neural network controller learnt from demonstrations was proposed. That was the first try to apply machine learning methods to the real HDM machine tasks. However, the method utilizes only the machine's low level sensors and experiments were performed in highly restricted conditions with limited variance. In real world, machines should learn from real data demonstrated by a professional operator and a controller must learn a robust representation which works even if the conditions change.

The primary goal of this work is to develop a controller robust to varying conditions such as loading distance, pile shape, weather and illumination. The goal is achieved by adopting Random Forest (RF) regression [4] as the controller model and using low-dimension deep vision features as complementary control inputs. These design choices are inspired by the recent works where visual features have been used in Learning from Demonstration (LfD) [5], [6] and decision trees as robust methods for classification and regression [7], [8], [9].

[1]Computing Sciences and [2]Automation Technology and Mechanical Engineering, Tampere University, Finland.

Fig. 1: Avant autonomous robotic wheel loader performing earth moving. Example videos in the supplementary material

**Contributions –** The main contributions of this work are:

- Vision is introduced as a powerful sensor modality for a HDM pile loading controller. Application specific deep Convolutional Neural Network (CNN) features are learnt using a Siamese architecture and a combination of cross-entropy and contrastive loss between successful and unsuccessful demonstrations. Deep visual features provide a strong complementary cue to the low-level signals (boom angle, bucket angle and hydraulic driving pressure).
- Random Forest (RF) regressor is introduced as a controller model for pile loading. RF maps the sensor signals to control commands using an ensemble of decision trees. RF provides robustness against changing conditions as compared to a shallow neural network used in the prior art (where it requires careful selection of demonstration).

The deep vision feature and RF regressor based controller is deployed to a robotic wheel loader to experiment loose material handling on a real scale machine; to the authors' best knowledge this is the first working controller based on vision and low level sensors on a moving robotic wheel loader.

## II. RELATED WORK

The majority of works on earth moving automation with heavy-duty machines are either heuristic or model based.

Filla et al. [10] propose four algorithms for parametric generation of end-effector motion trajectories based on the observed professional driver experiences. Fernando et al. [11] presented an admittance controller for autonomous loading from muck pile containing large rocks. The heuristic algorithm applies specific forces to the pile rather than following the desired trajectory. This allows to solve the excavation problem when the pile contains large rocks. Jud et al. [1] propose to solve the problem of autonomous excavation by utilizing the trajectory of end-effector force-torque. This approach allows to avoid arbitrary high forces compared to the algorithms making use of end-effector position trajectory. The authors also implemented a large-scale planner for continuous digging operation which is a problem that is not often addressed. All experiments were performed in a simulator. Sotiripoulos and Asada [2] propose autonomous excavation based on the power transmitted from the excavator to soil as a signal for adaptive excavation. By maximizing the output product of force and velocity the method enables the optimal bucket filling control. They demonstrated that the test prototype can execute excavation adaptively with respect to different soil conditions. However, the model was deployed only to a small-scale prototype with a single phase of excavation and without vision.

Several works learn the parameters of the desired actions from demonstrations. Dadhich et al. [12] present preliminary analysis of the data recorded during demonstrations. The authors fit linear regression models to the lift and tilt bucket commands recorded with a joystick, but the approach is not experiment in realistic cases. Fukui et al. [13] proposed an imitation-based approach to excavation motion planning which was implemented for a 1/10-scale excavation set-up. The authors collected examples of excavation motion for different appearances of the rock piles. The appearance was determined by geometrical features, particle size and color distributions. Thus, the database of excavation motion was connected to the pile visual appearances. Second, a neural network was trained to match the pairs. At the test time the motion, selected by the neural network from a list of predefined motions, was performed for the observed pile.

Halbach et al. [3] were the first to demonstrate a neural network controller learnt from demonstrations on the real-scale HDM machine. Unlike previous works, the controller was modeling the whole trajectory of the motion, including pile approaching and scooping. No vision was used in their system. The demonstrations were carefully selected and refined which does not reflect the real situation. The number of experiments was quite limited and conducted in similar conditions to the training demonstrations.

## III. APPROACH

Robot Learning from Demonstration (LfD) [14] is a paradigm for enabling robots to autonomously perform new tasks. In LfD an appropriate robot controller is derived from observations of human demonstrations. The aim is for robot capabilities to be more easily extended and adapted to novel situations, even by users without programming ability.

For real applications a controller must be robust to small changes in initial settings and work conditions. Robustness must be achieved despite of the challenges in LfD:

- Human demonstrations contain only a limited amount of input variation (e.g., collected during a single day at a single work site).
- The demonstrations vary in operating policies (see Figure 5).

The above restrictions make many machine learning techniques easily overfit to training data. Our work shows increased robustness to the above condition changes and therefore also better performance in the terms of success rate. Our approach is based on two important design choices: Random Forest regression is used as the controller model and visual information is used as additional sensor cue (which havs been used as features for control in multiple recent works [15], [6], [5]).

### A. Demonstration process and recorded signals

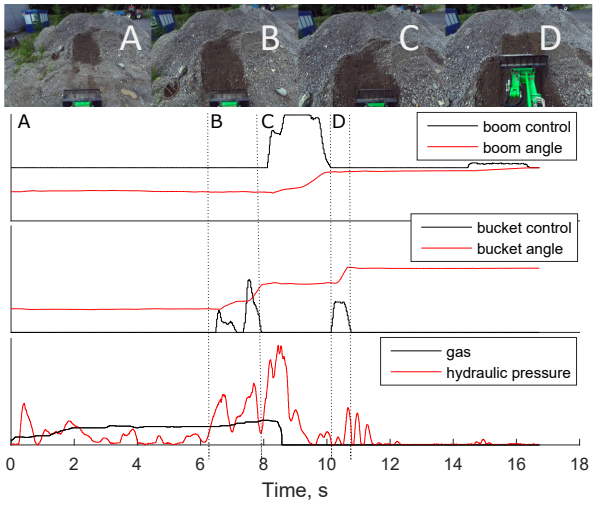



Fig. 2: A demonstration sample: egocentric view (top) and low-level sensor and control signals (bottom). The stages *approaching* (A), *preparing to scoop* (B), *soil impact* (C) and *scooping* (D) are added for illustration only.

The demonstrations were not generated by professional operators nor were they specifically instructed or selected to provide good training demonstrations. A sample demonstration is illustrated in Figure 2 (note that the different stages were not used but added only for illustration). Prior to each of the demonstrations, an operator manually positioned the boom at a good scooping height and the bucket in parallel with the ground level. After the demonstration, the bucket was unloaded to the same pile and the wheel-loader moved back to a certain distance and reset its pose. The distance to a pile was approximately the same. In total, 59 demonstrations were collected for controller training. In this work, no steering was used with an assumption that the pile is always in front of the machine. The pressure signal and visual features appear to indicate whether the pile has been reached. Another set of 20 unsuccessful scooping examples

(empty bucket) were collected to train an application specific visual features (Section III-B).

The following sensor and control signals were recorded during the human demonstrations:

- Boom joint angle $\phi$ and bucket joint angle $\psi$.
- Hydraulic drive transmission pressure $p$.
- Boom joint control $u_\phi$ and bucket joint control $u_\psi$
- Throttle or gas command $a$
- Eight-dimensional deep visual features $\mathbf{f_v}$, extracted from a set of five consecutive image frames $I_{i=1,2,...,5}$, details in section III-B.

In the reminder of this section, we explain visual features generation and random forest regression in more details.

### B. Deep visual features

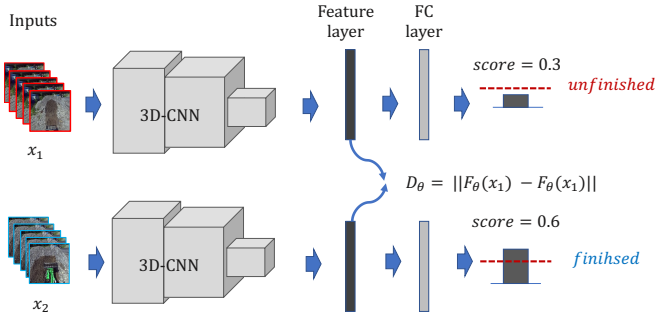



Fig. 3: A Siamese CNN architecture (weights are shared) for deep visual feature extraction. Input is a tensor of five frames $\vec{x}$ and the output is a 8-dimensional feature vector $\vec{f}_v$. The network is trained using a sample pairs, $\vec{x}_1$ and $\vec{x}_2$, from the successful and unsuccessful demonstrations and its based on a combination of a cross-entropy and contrastive loss.

In robotics applications, visual features are often trained in an architecture that enforces visual features to be connected to spatial information about the objects [16]. Recent works have shown that good performance can be achieved when training is based on visual examples of the final stages of the robot task [6]. This approach suits better to the pile loading task since it deals with loose material and it is hard to specify the ground truth about objects in the scene.

To provide a dynamic visual feedback, five consecutive frames are combined into a 3D visual tensor that is the input to a Convolutional Neural Network (CNN) architecture (Figure 3). The network was trained as a goal classifier which predicts if the pile loading task was successfully accomplished or not. For training 59 demonstrations represent success and 20 demonstrations failure cases. The second last layer of the architecture generates an eight-dimensional vector, which is used as the visual feature for control.

Training of the CNN architecture is based on a simple target: the network should learn to distinguish samples from the successful samples (bucket is full and task accomplished) and unsuccessful samples (bucket is empty or task is unfinished). For the main target the standard cross-entropy loss works well. However, in addition to the cross-entropy loss for classification, a contrastive loss [17] was added to constrain visual feature extraction. The contrastive loss is added to control the type of features extracted - the features within the positive examples should be similar and different from the negative samples. The contrastive loss has been used in one-shot learning tasks and metric learning tasks where feature distances become important [18].

All demonstrations are denoted as $Z$ where $Z_n$ are the negative (unsuccessful) and $Z_p$ positive (successful) samples. $F_\theta$ is the feature extraction network with parameters $\theta$. Let $x_1, x_2 \in Z$ be the input image sequences samples (see Figure 3). Then, the extracted feature representations are $f_{v1} = F_\theta(x_1)$ and $f_{v2} = F_\theta(x_2)$. The Euclidean distance between the two encoded features is defined as $D_\theta$:

$$D_\theta = \| f_{v1} - f_{v2} \|_2 = \| F_\theta(x_1) - F_\theta(x_2) \|_2 \quad (1)$$

The contrastive loss uses the Euclidean distance as:

$$L_c = \begin{cases} \max(0, m - D_\theta(x_1, x_2))^2, & \text{if } x_1 \in Z_n,\ x_2 \in Z_p \\ D_\theta(x_1, x_2)^2, & \text{if } x_1,\ x_2 \in Z_p \end{cases} \quad (2)$$

$x_1$ and $x_2$ are input into a shared-weights siamese network. If $x_1$ and $x_2$ belong to the positive class, the contrastive loss $L_c$ is set to minimize $D_\theta$; otherwise, parameter $\theta$ should be updated to increase $D_\theta$. Parameter $m$ ($m > 0$) is an empirical margin value based on the spring model analogy [17].

The final loss used to train the convolutional neural network is defined as:

$$L = \lambda_1 L_{ce}(G_\theta(F_\theta(x))) + \lambda_2 L_c, \quad (3)$$

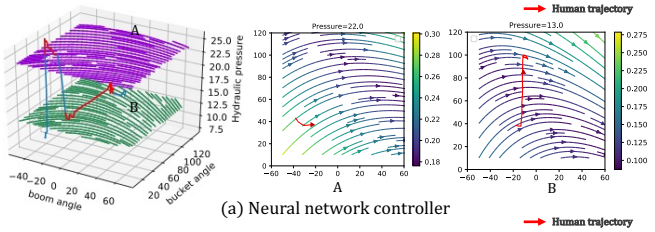
where $x \in Z$, $L_{ce}$ is the standard cross-entropy loss for task classification and $L_c$ is the above contrastive loss. $\lambda_1 = 0.6$ and $\lambda_2 = 0.4$ denote the weight of the losses.

Since the demonstrations are limited and data is unbalanced (59 videos for successful scooping task and 20 for failure cases), a state-of-the-art video frame interpolation method DAIN (Depth-Aware Video Frame Interpolation) [19] [1] was used to generate more training examples. To avoid overfitting and robustify against image noise, image data augmentation was applied during the training. The CNN visual feature extractor and Avant controllers were trained separately.

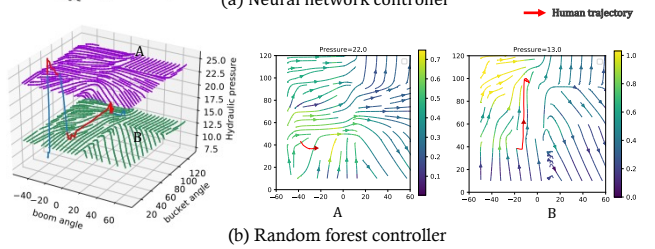
### C. Random Forest controller

In the prior work, a small shallow neural network controller (NNet) was proposed in [3]. It consists of a single hidden layer with 5 to 10 neurons, and a fully connected output layer for control outputs. The best results were obtained by training the network with the Levenberg-Marquardt algorithm. In our preliminary experiments, the neural network controller performed poorly as it was sensitive to changing conditions. In prior work [3], the demonstrations were carefully selected However, the demonstrations (operation and sensor data) in this work show more variation (see Figure 5). The supervised neural network with simple shallow structure is not capable of learning the complex

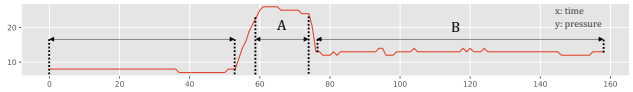
[1] code used: https://github.com/baowenbo/DAIN




(a) Neural network controller




(b) Random forest controller




(c) Hydraulic pressure changes (human operation)

Fig. 4: Comparison of the control vector fields of (a) a neural network controller and (b) random forest controller trained with the same demonstrations. x-axis is the bucket angle and y-axis is the boom angle. The arrow color denotes the speed (the value bars on the right of each image). The pressure changes are shown in (c). For illustration the following stages are used: A (approaching, the average pressure is 22.0), B (soil impact, avg. pressure 13.0). Red arrow denotes a single demonstration in the training set.

behaviour corresponding to high variance observations, but tends to converge to average behavior.

To overcome the limitations of the neural network controller, we propose to use Random Forest (RF) regression [4] instead. RF regression is more robust to the data intrinsic ambiguities, when different output values might be associated with the same input values [7]. In certain low-dimension regression tasks, random forest is still superior to neural network [20][21]. This is very important in our case with large variations in demonstrations. The works such as [22], [23] and [24] have shown how RF regression is capable of learning feature representations. In our case, each tree learns only from partial set of features and therefore can learn important cues about the action.

In this work, the random forest $\mathcal{F}_{rf}$ is a collection of regression trees (100 trees and maximum depth is 20):

$$\mathcal{F}_{rf} = \{\mathcal{T}_m(\Theta_m), m = 1, 2, ..., m\}$$

$\Theta_m$ denotes the parameters of each tree $\mathcal{T}_m$. Given the input signal observation $v = \{\phi, \psi, p, f_v\}$, the regression results produced by $\mathcal{F}_{rf}$ is:

$$\mathcal{F}_{rf}(v) = \frac{1}{m}\sum_{i=1}^{m}\mathcal{T}_m(v; \Theta_m) \tag{4}$$

In this work for regression problem, the control prediction is an average of all prediction results produced by the trees. Let's denote the training set as pairs $\{v_i, y_i\}^N, i = 1, 2, ..., N$, where $y_i = \{u_{\phi i}, u_{\psi i}, a_i\}$ is the output control commands. The training of $\mathcal{F}_{rf}$ is performed as following:

- Draw a bootstrap data $D_{bs}$ from the training set $v_i, {y_i}^N$.
- Grow a meta regression tree $\mathcal{T}_m$ to the bootstrapped data $D_{bs}$, fit each regression tree until the maximum depth is reached.

Each tree $\mathcal{T}_m$ is trained as following:

- Randomly select $n$ features from the $k$ features ($n < k$).
- Pick the best variable split-point among the $n$.
- Split the node into two child nodes.

Each tree $\mathcal{T}_m$ is trained by minimizing a sum of squared error function $SSE$:

$$\min(SSE) = \min(\sum_{i=1}^{m} ||\mathcal{T}_m(v_i; \Theta_m)) - y_i||_2^2) \tag{5}$$

Then the whole random forest $\mathcal{F}_{rf}$ is trained by minimizing:

$$\min(SE) = \min(||\mathcal{F}_{rf}(v_i)) - y_i||_2^2) \tag{6}$$

Figure 4 compares the control policies of NNet and RF controller. The NNet policy is smooth, but it is actually an average behavior in the training data while RF learns more complex policy that corresponds to the real case better. For example, when the pressure is $\approx 13.0$, the trajectory generated by the neural network controller performs differently than the human demonstration (the bucket angle is supposed to be lift up). The RF controller action vectors correspond to the human example trajectory.

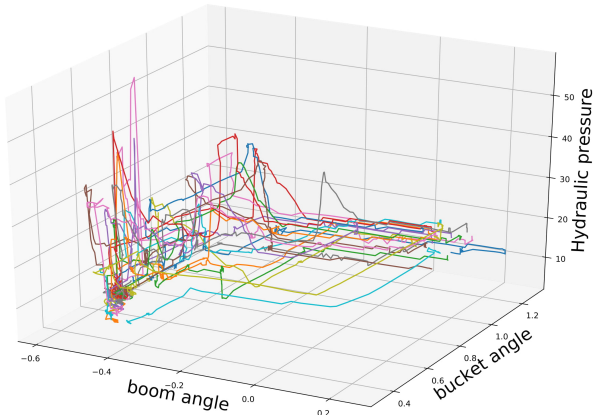



Fig. 5: 20 different human demonstration trajectories in sensory data space.

## IV. EXPERIMENTS

### A. Experimental setup

The autonomous scooping was implemented on a robotic wheel-loader, so called GIM machine. It has the mechanics of a commercial wheel loader (Avant 635) and power transmission and controllers are custom made at Tampere University. The bucket is positioned in vertical plane by two joints, the boom joint and the bucket joint, and in horizontal plane by drive (throttle/gas) and articulated by a

frame steering mechanism[2]. The GIM machine is equipped with various sensors including, for example, GNSS (Global Navigation Satellite System), wheel odometry and IMU (Inertial Measurement Unit). We used a ZED stereo camera to get the visual feedback. The ZED camera images of $2560 \times 720$ resolution were captured at the frame rate of $15 fps$. Instead of depth the left RGB image was used by the control algorithm since the feature extraction CNN was pre-trained with RGB data. Hydraulic drive transmission pressure sensors, which reflect forces generated by the wheels, were also used. The experiments were performed at the outdoor test-site (see Fig. 1).

The control system is composed of multiple layers. In the very low level (digital and analog I/O and CAN), industrial micro-controllers implement the power management and basic safety functions. In the PC control level, a target PC runs Realtime Simulink models, which run real-time tasks such as localization. Sub-systems communicate low level sensor data and control commands via UDP protocol running on a Jetson AGX Xavier (8-Core ARM v8.2 64-bit NVIDIA Carmel CPU and 512-core NVIDIA Volta GPU with 64 Tensor Cores) on-board. All the data collection, learning and closed-loop control are implemented on Jetson. Overall system performance was about 8Hz, but it was reduced to 3Hz to make execution of commands more feasible for Avant.

### B. Performance metrics

Fig. 6: Manually defined performance classes from left to right: full, half-full and empty (failed) bucket fill.

The bucket load after each test sequence was manually classified to one of the following three classes: *full*, *half-full*, and *empty* (Figure 6). With $N$ being the total number of test runs in one experiment, we report the percentage of runs resulted in full, half-full, and empty buckets correspondingly: $M_{full} = \frac{N_{full}}{N} \cdot 100\%$, $M_{half} = \frac{N_{half}}{N} \cdot 100\%$, $M_{empty} = \frac{N_{empty}}{N} \cdot 100\%$. We also report a total success rate $R = M_{full} + M_{half}$. The tests runs were conducted on several different days over a period of 30 days to observe performance on different weather and light conditions. The distance to the pile was varied and the wheel loader was positioned approximately toward the pile. The user demonstrations for regressor training were collected during a single sunny day.

[2]Avant 635 is a multi-purpose loader and also used for pallet loading, thus the boom (manipulator) comes with extra degree of freedom a telescopic (prismatic) boom, which is not used in this study, since it is not common in earth moving.

We illustrate the main experimental findings in three different cases:

1) We compare performance of the proposed RF regressor to the previously proposed neural network regressor (NNet) [3] to verify RF regressor's performance.
2) We show that adding visual cues further improves performance of the RF controller.
3) We further experiment robustness of the RF controller against variation in the distance to the pile.

### C. Case 1: RF regressor vs. NNet

TABLE I: Performance metrics of autonomous bucket filling over 50 trials at varying distance and on five different days over a period of 30 days. Only the low-level sensors are used without vision. Note that a smaller number for $M_{empty}$ is better.

.

| | $M_{full}$ | $M_{half}$ | $M_{empty}$ | $R$ |
|---|---|---|---|---|
| NNet [3] | 0% | 12% | 88% | 12% |
| RF | **52%** | **28%** | **20%** | **80%** |

In this experiment the performance of the proposed RF controller was compared to the neural network controller by Halbach et al. [3]. Both controllers were trained using the same demonstrations and the same low-level sensor signals (boom angle $\phi$, bucket angle $\psi$, and hydraulic drive transmission pressure $p$). In this work we aim at an extensive testing with the following varying conditions:

- Distance to the pile: $\sim 4.0\ m$ - $\sim 5.5\ m$.
- Different weather conditions and different time on multiple days.
- Two types of working area soil: gravel and concrete.

Each controller was tested in total 50 times with varying conditions. The performance metrics are shown in Table I, where the RF controller improves the total success rate $R$ by 68% as compared to neural network regression. NNet fails in most of the cases, i.e., rising the boom and bucket too early before reaching the pile which results in an empty bucket. A possible reason is that the neural network did not learn to associate the pressure signal with distance to the pile and only learnt an average trajectory of the training set. The successful performance of the RF controller even with varying conditions can be explained by the power of randomization in feature selection. Each tree learns only a single "aspect" of the solution, e.g., the pressure cue, and therefore even if many trees fail on novel cases a substantially large number of them has the correct aspect and they therefore vote for correct behaviour.

### D. Case 2: Effect of deep visual features

In this experiment, the goal was to study the effect of visual features on controller's performance. The following settings were tested:

- Distance to the pile: $\sim 4.3\ m$ - $\sim 4.8\ m$

TABLE II: Comparing the complementary information of deep visual features and the low-level sensor signals. The numbers are computed from 10 trials.

| | $M_{full}$ | $M_{half}$ | $M_{empty}$ | $R$ |
|---|---|---|---|---|
| RF w/ sensors | 30% | 10% | 60% | 40% |
| RF w/ vision | 40% | **30%** | 30% | 70% |
| RF w/ sensor+vision | **60%** | 20% | **20%** | **80%** |

- The experiments were conducted during the same time of day on multiple days.
- Working area soil: concrete.

The results of the second experiment are shown in Table II. Vision clearly improves the success rate of bucket filling as the failure rate drops from $60\%$ to $20\%$. However, it is also striking how well the vision-only controller works - it performs better than the low-level sensor controller indicating that visual cues learnt using the proposed architecture are strong for learning autonomous pile loading. The results indicate that vision provides complementary cues to the low-level signals as their combination performs clearly the best.

### *E. Case 3: Robustness to distance variation*

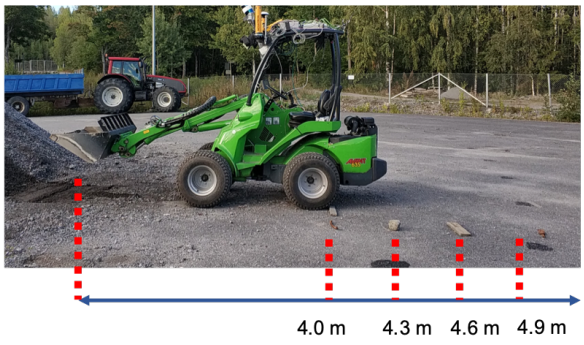



Fig. 7: Experimental set-up for distance variation $d$ testing: four initial positions.

The controller's ability to adapt to different initial locations proved to be crucial in our experiments. If the pile is detected too early, the machine will start scooping before reaching the pile which will lead to an empty bucket. In Case 3 we will have a closer look on the Case 1 results and also add the RF model with vision. Note that the 50 test runs were done starting from multiple distances to the pile (Figure 7).

The results for different distances are summarized in Figure 8 (NNet is omitted due to its poor performance as its success rate $R = 12\%$).

The benefit of using visual features in addition to the low-level signals is obvious, the deep visual feature have clear positive effect on full-bucket performance: for all runs at the distances $4.0\ m$, $4.3\ m$, and $4.6\ m$, $Vision\_RF$ controller (the controller using low-level sensors and vision) and $RF$ controller (the controller using only sensory data) both achieved well task performance $R = 100\%$. However, for $Vision\_RF$ controller, it obtained better performed full-bucket rate $M_{full} = 100\%$; while for $RF$ controller, the controller's full-bucket rate are significantly decreased ($M_{full} = 60\%, d = 4.0; M_{full} = 60\%, d = 4.3; M_{full} = 70\%, d = 4.6$).

At $4.9\ m$, scooping failures start to occur to both controllers. The runs on the furthest distance were conduceted late in the evening so also the amount of light is affecting the results. Although the extracted visual features is robust to certain environment condition changes, it is not strong enough to handle the unseen conditions (e.g. the evening light condition). In total, $Vision\_RF$ controller has success rate $R = 85\%$ and $RF$ controller has success rate $R = 75\%$.

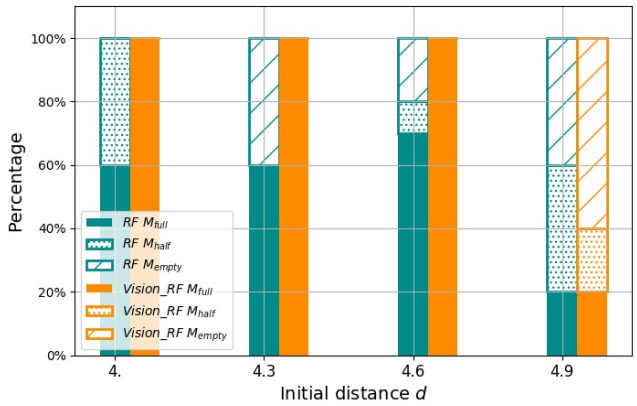



Fig. 8: Task accomplish histograms for the different controllers (RF w/ low-level sensors and RF w/ sensors+vision ) from four different starting distances.

Overall, the proposed RF controller showed robust results and a significant improvement over the prior art (NNet). Within certain range ($d \in [0\ m, 4.6\ m]$), the deep visual features improved the pile loading task performance. However, due to the limited data, the deep feature still have the shortcoming for unseen conditions (e.g. the evening light condition and extremely far distance).

## V. CONCLUSION

In this work, a novel controller was proposed for autonomous pile loading. The controller parameters were optimized using the LfD framework. The controller benefit from two important design choices: the main model is a Random Forest (RF) regressor that provides robust regression and it adopts deep vision features which are constructed from the same training data and which provide strong complementary cue to low-level signals. In the experiments, the RF regressor was robust against several variations (distance to the pile, type of soil, weather and illumination) while the prior art often failed. In our future work, we will improve the robustness of the visual features, and adopt Reinforcement Learning to further improve the LfD pre-trained controller to find even more optimal and robust parameters.

## REFERENCES


[1] P. L. D. Jud, G. Hottiger and M. Hutter, "Planning and control for autonomous excavation," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2151–2158, 2017.

[2] F. E. Sotiropoulos and H. H. Asada, "A model-free extremum-seeking approach to autonomous excavator control based on output power maximization," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1005–1012, 2019.

[3] E. Halbach, J.-K. Kämäräinen, and R. Ghabcheloo, "Neural network pile loading controller trained by demonstration," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[4] K. D. C. Andrew R. Webb, *Statistical Pattern Recognition*. Wiley, 2011.

[5] A. Singh, L. Yang, K. Hartikainen, C. Finn, and S. Levine, "End-to-end robotic reinforcement learning without reward engineering," *Robotics: Science and Systems*, 2019.

[6] M. Vecerík, O. Sushkov, D. Barker, T. Rothörl, T. Hester, and J. Scholz, "A practical approach to insertion with variable socket position using deep reinforcement learning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[7] A. Criminisi, J. Shotton, and E. Konukoglu, *Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning*, vol. 7, pp. 81–227. NOW Publishers, 2012.

[8] B. Lakshminarayanan, D. Roy, and Y. Teh, "Mondrian forests: Efficient online random forests," *Advances in Neural Information Processing Systems*, vol. 4, 06 2014.

[9] Z.-H. Zhou and J. Feng, "Deep forest: Towards an alternative to deep neural networks," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3553–3559, AAAI Press, 2017.

[10] B. F. R. Filla, M. Obermayr, "A study to compare trajectory generation algorithms for automatic bucket filling in wheel loaders," tech. rep., Volvo Construction Equipment, Fraunhofer ITWM, Lund University, 2014.

[11] H. A. H.A. Fernando, J. Marshall and J. Larsson, "Towards controlling bucket fill factor in robotic excavation by learning admittance control setpoints," in *Springer Proceedings in Advanced Robotics 5*, pp. 35–48, 2018.

[12] F. S. S. Dadhich, U. Bodin and U. Andersson, "Machine learning approach to automatic bucket loading," in *24th Mediterranean Conference on Control and Automation (MED)*, 2016.

[13] M. N. R. Fukui, T. Niho and M. Uetake, "Imitation-based control of automated ore excavator to utilize human operator knowledge of bedrock condition estimation and excavating motion selection," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5910–5916, 2015.

[14] T. Osa, J. Pajarinen, G. Neumann, J. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," *Foundations and Trends in Robotics*, 2018.

[15] C. Schenck and D. Fox, "Visual closed-loop control for pouring liquids," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2629–2636, 2017.

[16] C. Finn, X. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel, "Deep spatial autoencoders for visuomotor learning," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 512–519, 05 2016.

[17] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. 1735–1742, 2006.

[18] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, vol. 2, 2015.

[19] W. Bao, W.-S. Lai, C. Ma, X. Zhang, Z. Gao, and M.-H. Yang, "Depth-aware video frame interpolation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[20] Z.-H. Zhou and J. Feng, "Deep forest: Towards an alternative to deep neural networks," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3553–3559, AAAI Press, 2017.

[21] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, "Do we need hundreds of classifiers to solve real world classification problems?," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3133–3181, 2014.

[22] Y. Kong and T. Yu, "A deep neural network model using random forest to extract feature representation for gene expression data classification," *Scientific reports*, vol. 8, no. 1, p. 16477, 2018.

[23] A. Tang and J. T. Foong, "A qualitative evaluation of random forest feature learning," in *Recent Advances on Soft Computing and Data Mining*, pp. 359–368, Springer, 2014.

[24] C. Vens and F. Costa, "Random forest based feature induction," in *IEEE 11th International Conference on Data Mining*, pp. 744–753, 2011.