AGPNet - Autonomous Grading Policy Network

Chana Ross[1], Yakov Miron[1,2], Yuval Goldfracht[1], Dotan Di Castro[1]
{Chana.Ross, Yakov.Miron, Yuval.Goldfracht, Dotan.DiCastro}@bosch.com,

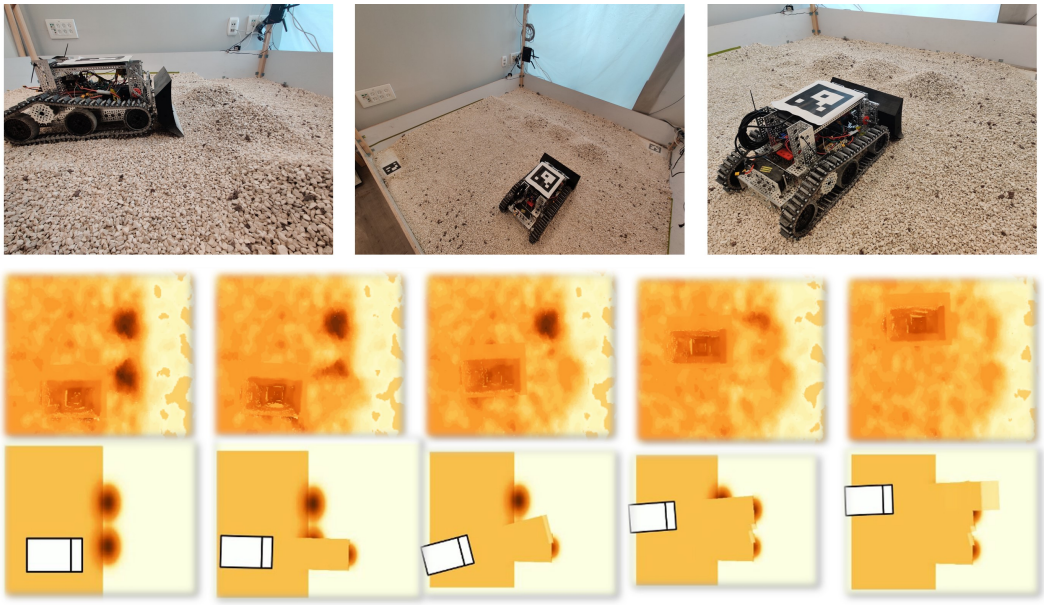


Fig. 1. A comparison of a trained agent executing autonomous grading on a **simulated environment** and a **real-world scaled environment**. **Top Row:** RGB images of our experimental setup (see Section III) showing the scaled dozer prototype facing the sand piles. **Middle and Bottom Rows:** A selection of heightmaps that depict actual states extracted from our real-world scaled environment and simulated environment respectively. Dark blobs indicate the sand piles while the left column represents the initial state of both environments.

***Abstract*—In this work, we establish heuristics and learning strategies for autonomous control of a dozer grading an uneven area studded with sand piles. We formalize the problem as a Markov Decision Process, design a simulation which mimics dozer-soil interactions and finally, compare our simulator to a real-world scaled environment. We use methods from reinforcement learning, behavior cloning and contrastive learning to train a hybrid policy. Our trained agent, AGPNet, reaches human-level performance and outperforms current state-of-the-art machine learning methods for the autonomous grading task. In addition, we show that our agent is capable of generalizing from random scenarios to unseen real world problems.**

I. Introduction

The off-road autonomous driving industry has attracted increasing interest in the past two decades due to shortage in experienced drivers as well as rising demand in the construction industry. Previous work in the field has mainly focused on obstacle avoidance [4], [11], optimal trajectory planning [12], [6], [11] and traversability [13]. The autonomous grading task was first tackled from a path-planning perspective by [2]. This pioneering study was the first to directly address the autonomous grading problem. In their work, [2] implemented a rule-based approach, where, given a large sand pile, the system selects the goal points the agent needs to reach and the grading leg it needs to perform. After the agent aggregates several of these legs, the pile is graded and the task is considered done. While this approach relies on rule-based heuristics, recent successes with machine learning methods have demonstrated the possibility of automating and optimizing such complex problems.

In this work, We focus on autonomous path planning for construction site vehicles. Specifically, we discuss the task of grading a given area with a number of sand piles. In this task, the dozer is confronted with an uneven terrain (Fig. 1) and is required to level the ground, in a minimal amount of time, to a predefined target height. We solve this problem using a hybrid approach which combines Reinforcement Learning (**RL**), a sub-field of machine learning, Behavior Cloning (**BC**) and Contrastive Learning (**CL**). In addition, We simplified the learning process by utilizing domain knowledge regarding the action space and added a prior on the initial action distribution.

As we demonstrate herein, our model is capable of training on a random scene, and then generalize to a more complex realistic problem. To validate our method, we created a simulation for training and evaluating our models, which includes all the important interactions between the dozer and the soil. In addition, we've built an scaled prototype environment in order to validate our methods on real-world data (see III).

[1]Bosch Center for Artificial Intelligence, Haifa, Israel
[2]ANFSL, The Hatter Department of Marine Technologies, University of Haifa, Israel

Our main contributions are: (1) We provide an end-to-end pipeline for training autonomous dozers that combines **BC** and **RL** for improved robustness, enhanced performance and reduced sample complexity. Here, we implement a hierarchical architecture in which high-level trajectory planning is learnt and low-level action control is performed. (2) We establish a **RL** environment simulator for the earth-moving dynamics and the interaction between the dozer and the soil. Using this simulator, we train a **RL** agent for the autonomous-grading task. (3) We validate the simulator using a scaled prototype environment and compare heightmaps generated by our simulator to those taken using a real depth camera.

II. Problem Formulation and Preliminaries

Our goal is to create an optimal policy for a bulldozer preforming a grading task, where given an initial grading area which contains a number of sand piles, the dozer must find an optimal trajectory to flatten the sand piles to a predetermined target height. We solve the problem using a combination of Behavioral Cloning (**BC**), ContrastiveLlearning (**CL**) and model-free Reinforcement Learning (**RL**) techniques. In our suggested method, we split the autonomous grading task into three independent sub-tasks: the initialization task, the continuous task, and the edge task (see Fig. 2). In addition, we formulate all tasks as a POMDP [10] where both the initial and target heightmaps are given.

A. Partially Observable Markov Decision Processes

A Partially-Observable Markov Decision Process (POMDP; [10]) is comprised of the tuple $(\mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{P}, \mathcal{R})$. While a state $s \in \mathcal{S}$ contains all the required information, in practice, agents are presented with partial information regarding the environment i.e observation $o \in \mathcal{O}$. After the agent selects an action $a \in \mathcal{A}$, the system transitions to the next state s' based on the transition kernel $P(s'|s,a)$ and the agent is provided with a reward $r(s,a)$.

The goal of an agent is to learn a behavior policy π^* (stochastic or deterministic) that maximizes the cumulative reward to go according to $\pi^* = \arg\max_{\pi \in \Pi} \mathbb{E}[\sum_{t=0}^{T}(\gamma^t r_t)]$.

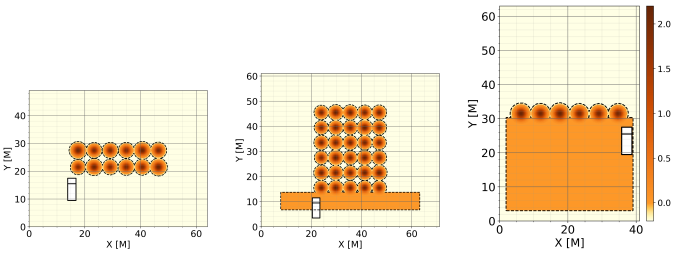
B. Problem Formulation

In this section, we will describe all of the POMDP components as they are reflected in the suggested solution.

State: The state s_t includes the target area size, dozer's location within this area, the full dozer trajectory up until the current time point, and the relative heightmap of the target area (denoted as $\delta_{H_t} = H_t - H_{des}$ where H_t is the current heightmap and H_{des} is the target heightmap).

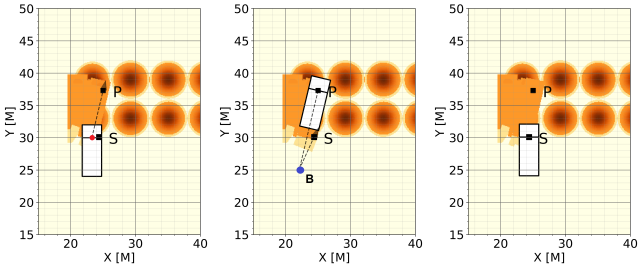
Observation: The observation o_t is comprised of an *EGO view* heightmap i.e a bounding box view around the current location of the dozer (derived from the full state's heightmap). While the state's heightmap dimensions can vary we keep the ego view's size fixed to make training simpler.

Action: While control over the dozer can be executed at a low level resolution, i.e speed and rotation, we chose to formulate our action space at a higher level selecting way-points to which the dozer should move. Each action a_t is parametrized as a way-point tuple $a_t = (p_t, s_t)$ where p_t is the first way-point reached by a forward movement and s_t is the starting point of the next action. Each action a_t is sampled from the two policy distributions over the FOV pixel map of the agent ($\pi \in R^{2\times\hat{m}\times\hat{n}}, \hat{m} = \frac{m}{2^N}, \hat{n} = \frac{n}{2^N}$), where $N = 3$ is the down-sampling factor, for reduced state and action space dimensions. Once these points are selected, we can continue and generate the low-level actions that form the entire trajectory of the vehicle as seen in figure 3 and similarly to [2]. Since a dozer typically drives in straight lines to avoid slippage, we chose to define these actions as a continuous movement in the dozer's body axis: δ_x for translation and δ_ψ for rotation.



(a) Initialization (b) Continuous (c) Edge

Fig. 2. The first state for each sub problem. In the **initial** scenarios, there is no graded area and a few rows of sand are dumped. The dozer is required to create an incline and reach the target height, H_{des}, if possible. Meanwhile, the dumper will add more sand piles for grading in front of the initial ones, creating more rows of sand. In the **continuous** scenarios, the agent is located at H_{des}, i.e., on top of the previously graded area, and sand piles are continuously being added in the vicinity of the graded area. The task is to constantly grade them to H_{des}. The main difference from the previous problem is that the piles are dumped on top of H_{des} and the dozer needs to push it forward, thus enlarging the area in which $H = H_{des}$. Finally, in the **edge** scenarios, the final row of sand piles is dumped, most of the area is already graded, and the sand leftovers need to be cleared. The dozer must create a decline to flatten the sand and then smooth the graded area.



(a) original state (b) state after p (c) state after s

Fig. 3. Example of MDP actions (p, s) and the trajectory between these actions. In 3a, the initial position is the red dot. The action's order is as follows: (i) From origin rotate to face p. (ii) Drive forward to p. This action has the greatest value w.r.t. the received reward, as it is the only one that grades sand. (iii) Reverse back to B (blue dot in 3b). (iv) Rotate to face next s. (v) Drive forward to next s. (vi) Rotate to face the sand piles (3c).

Reward: The grading task involves many objectives, which need to be reflected in the reward function. In our environment, the reward is multi-modal. One mode takes *time* as an objective, so only discounting over the horizon might collapse to the trivial solution (minimal time while not completing the task or not touching the sand at all in the fatal case). In the general case, the optimal agent will *complete the task*, i.e., reach the target surface, H_{des}, not leave sand

piles/bumps in the area i.e., will remove the *maximum volume*, and grade sand in every leg, i.e., will minimize the legs in which reverse/rotation actions are selected. Moreover, upon *task completion*, the agent gets a large reward, and if not accomplished, the agent receives a large negative reward. See section III-B. The multi-objective reward function is: $R_t = \lambda_v * f_v - \lambda_t * f_t + \lambda_h * f_h + \lambda_d * \mathbb{1}_{is_{done}} - \lambda_f * \mathbb{1}_{is_{failed}}$, where the current volume removed f_v (calculated as the sum over $H_T - H_{init}$, current height removed f_h and time spent f_t on executing the action. All the f_i functions and λ_i coefficients of the specific rewards were tuned during the hyper-parameter search.

III. Method

In order to tackle the problem of autonomous grading we focus our efforts on several fronts: (1) implementing a realistic simulation environment (2) creating a rule-based heuristic policy for autonomous grading (3) exploring machine learning methods that can succeed in solving this challenging task. (4) Validating our simulation and policy on a scaled prototype environment

A. Simulation

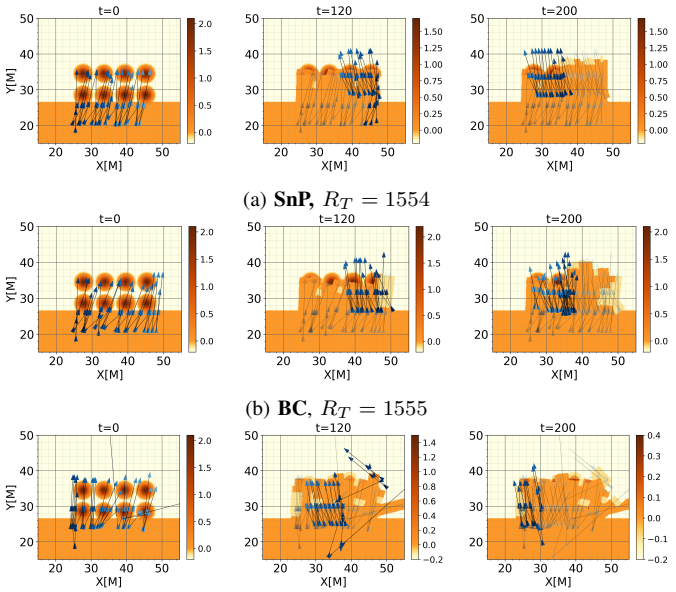
The movement of the soil due to a dozer's action is not trivial and can be *simulated* using different techniques, each one capturing different aspects of the full interaction [7], [9], [5]. We aim to create a computationally inexpensive simulation, while taking into account key aspects of the environment which are needed to estimate an optimal policy. These aspects include the interaction between the dozer and the soil as well as the dozer's behaviour such as velocity change due to torque.. In our simulation, each sand pile is modeled as a Multivariate Gaussian Distribution with two variables (x, y). These variables are the cartesian coordinates of the height map, both taken from an i.i.d normal distribution,

$$f(x,y) = \frac{V}{2\pi\sigma_x\sigma_y} * \exp\left(-\frac{1}{2}[(\frac{x-\mu_x}{\sigma_x})^2 + (\frac{y-\mu_y}{\sigma_y})^2)]\right),$$

where $f(x, y)$ is the height of the soil at each point, $V[cm^3]$ is the volume, and $\sigma_x[cm], \sigma_y[cm]$ define the footprint of the sand pile. For example, given a volume V, as σ_x and σ_y grow, the piles height is reduced and footprint grows.

B. Heuristic Policy

Throughout our work, we used the rule-based heuristic policy, denoted as **SnP** ("start point" and "push point"), inspired by [2] as an experienced agent and a good baseline comparison. This policy is based on a human expert and mimics the expert's behaviour. As done in the trained policies, we used two levels of action here: The way-point planner chooses the destination point and then the origin (p_t, s_t) and the path planner creates 6 low-level actions, as shown in Figure 3. Unlike our learnt algorithms, the high-level actions are chosen using a combination of classical detection algorithms for sand-pile detection and search heuristics.



(a) **SnP**, $R_T = 1554$

(b) **BC**, $R_T = 1555$

(c) **RL+BC+CL**, $R_T = 1556$

Fig. 4. Comparison between trajectories from different policies for the continuous problem. Each triangle is another action and the lines indicate the paths between these actions. Each row shows a different policy (**SnP**, **BC**, **RL+BC+CL**) and each column a different stage of the episode. R_T is the total reward.

C. Scaled Prototype Environemnt

To validate our simulation, we created a **scaled prototype environment** at a $1:9$ scale compared to a real dozer. We built a sand box with an RGBD sensor that provides both heightmaps and agent locations within a global coordinate system. The prototype dozer interacts with the sand, and the height-maps can be recorded throughout the episode and be post-processed with the simulation of the same scenario. Figure 1 (center image) shows the experimental setup with *(i)* a sand box filled with sand prior to grading, *(ii)* the prototype dozer, and *(iii)* the localization system. ArUco [8] markers are used to localize the dozer and calibrate the camera w.r.t world coordinates. The dozer is controlled by 3 motors: two control the tracks and the third controls the blade. The second row in Fig. 1 shows the output of the experimental setup.

D. Policy Model

For all the methods, we utilized a deep neural network for estimating the policy for each state. Our models have an embedding layer followed by a number of convolution layers and a fully connected layer at the end. We used Yolo lite [3] or Resnet [1] architectures as a baseline and assumed a discrete action space. To ease the learning process, and improved robustness, we added a prior that the agent will initially prefer to move within its near vicinity, by adding a spatial Gaussian layer. Finally, a softmax layer for each sub-action was calculated to produce a distribution over the pixels.

IV. Experiments

To demonstrate our method and the robustness of our algorithms, we compared them on three types of problems,

scenario type	metric	BC	BC+CL	RL	RL+BC	RL+BC+CL	RL+CL	SnP
Init	**volume** ↓	0.63	21.97	1.03	0.57	0.25	**0.07**	0.36
	height ↓	2.06E-04	7.20E-03	2.71E-04	1.76E-04	**7.12E-05**	2.19E-05	1.15E-04
	time ↓	5024	6566	14464	14326	12975	11140	**3278**
	reward ↑	1478	930	1714	1916	2687	3132	1934
Continuous	**volume** ↓	15.04	46.53	0.70	0.36	0.29	0.16	**0.12**
	height ↓	4.10E-03	1.28E-02	1.87E-04	**9.92E-05**	8.63E-05	4.98E-05	3.74E-05
	time ↓	6232	7282	11455	15775	12857	8864	**3451**
	reward ↑	1584	712	1681	2910	2271	3123	2585
Edge	**volume** ↓	0.25	4.65	1.22	0.42	0.30	0.10	**0.05**
	height ↓	7.74E-05	1.81E-03	3.49E-04	1.20E-04	**8.89E-05**	3.39E-05	1.42E-05
	time ↓	4657	5670	10981	12065	5993	8468	**3094**
	reward ↑	939	475	1624	2447	**3690**	3512	2791

TABLE I. All the results for our algorithms, including the **SnP** heuristic, **BC**, **RL** and hybrid methods. Our hybrid methods achieve better results on the main metric (height) and the overall reward. Results are mean over 50 i.i.d. runs.

as outlined in Figure 2 and explained in Section II. We used 3 different algorithms: **BC**, **CL**, **RL** as building blocks for 8 different hybrid models and focused our comparison on 4 metrics. The dataset used to train the **BC** policies included 150 episodes, each with a range of states drawn from our simulator. For the purpose of evaluation, we ran 50 runs for each scenario type generated from the same distribution and compared the mean result for each metric. Each initial state had a different number of sand piles, set up in a lattice format, and the dozer was positioned facing the piles. All the algorithms were calculated on the same scenarios to ensure a fair comparison. Autonomous off-road planning is complex and, specifically, the grading assignment does not have classic solvers for comparison. We, therefore, compared our results to the **SnP** heuristic that is based on [2], who used experienced expert drivers and mimicked their behaviour.

We show our results on all the metrics in Table I. The results of the majority of our approaches are on par with the baseline heuristic. We found that our combined algorithms approach (**RL+BC+CL**) outperforms the heuristic in terms of important metrics and overall reward. In our approaches, the agent learns from experienced rule-based algorithms similar to other **BC** models but also trains on online policies allowing for exploration. In addition, we enhanced the policies' ability to detect important features in the state space by adding a **CL** loss (**CL**), and the final Gaussian masking layer in the policy ensures our agent does not explore irrelevant areas. In Figure 4, we show a comparison between the trajectories of 3 agents (**SnP**, **BC** and **RL+BC+CL**). As the **BC** agent is able to generalize to episodes it never saw, it successfully follow the pattern of the **SnP** heuristic. The **RL+BC+CL** trajectory has less actions and manages to grade more soil at an earlier step (see left column). Moreover, this policy reaches an overall higher reward and lower final height (better agent).

V. Discussion

We here present AGPNet, an end-to-end pipeline for autonomous grading using a dozer. First, we formulate the problem as an *MDP* and use **RL** and **BC** algorithms to solve the problem. Second, we create a light yet detailed simulation for training algorithms and suggest a new and innovative approach for simulating earth-moving vehicles and their interaction with the soil. We prove the validity of our simulation with a real prototype dozer and show how height-maps from the real dozer are comparable to the ones from our simulation. Last, we train multiple policies and show that combining different **RL** and **BC** approaches with a high level of detection training such as **CL** achieves on par results with the heuristic and generalizes in more complex scenarios. Our method is ideal for tasks where a vehicle has an interaction with the soil that effects the environment and changes the optimal sequence of actions. It can also be used in other construction vehicles where the way-point planning is complex but the low-level actions can be defined using simple rule-based methods.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, 2016. 3

[2] Masami Hirayama, Jose Guivant, Jayantha Katupitiya, and Mark Whitty. Path planning for autonomous bulldozers. *Mechatronics*, 2019. 1, 2, 3, 4

[3] Rachel Huang, Jonathan Pedoeem, and Cuixian Chen. Yolo-lite: A real-time object detection algorithm optimized for non-gpu computers. *2018 IEEE International Conference on Big Data*, 2018. 3

[4] A. Kelly, A. Stentz, O. Amidi, M. Bode, D. Bradley, A. Diaz-Calderon, M. Happold, H. Herman, R. Mandelbaum, T. Pilarski, P. Rander, S. Thayer, N. Vallidis, and R. Warner. Toward reliable off road av operating in challenging environments. *I. J. Robotic Res.*, 2006. 1

[5] Wooshik Kim, Catherine Pavlov, and Aaron M Johnson. Developing a simple model for sand-tool interaction and autonomously shaping sand. *arXiv:1908.02745*, 2019. 3

[6] H. Jayakody M. Swift and M. Whitty. Path planning for multi-object push problems in continuous domain. *IFAC-PapersOnLine*, 2019. 1

[7] M. Pla-Castells, I. Garcıa-Fernández, and R. J. Martınez. Interactive terrain simulation and force distribution models in sand piles. 3

[8] Francisco J Romero-Ramirez, Rafael Muñoz-Salinas, and Rafael Medina-Carnicer. Speeded up detection of squared fiducial markers. *Image and vision Computing*, 76:38–47, 2018. 3

[9] A. Sauret, N. Balmforth, C. Caulfield, and J. McElwaine. Bulldozing of granular material. *Journal of Fluid Mechanics*, 748:143 – 174, 2014. 3

[10] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018. 2

[11] Yusheng Xiang, Kailun Liu, Tianqing Su, Jun Li, Shirui Ouyang, Samuel S Mao, and Marcus Geimer. Extension of bim using ai: a multi working-machines pathfinding solution. *arXiv:2105.06635*, 2021. 1

[12] Yanfu Zhang, Wenshan Wang, Rogerio Bonatti, Daniel Maturana, and Sebastian Scherer. Integrating kinematics and environment context into deep inverse reinforcement learning for predicting off-road vehicle trajectories. *arXiv preprint 1810.07225*, 2018. 1

[13] Zeyu Zhu, Nan Li, Ruoyu Sun, Donghao Xu, and Huijing Zhao. Off-road autonomous vehicles traversability analysis and trajectory planning based on deep inverse rl. *Intelligent Vehicles Symposium (IV)*, 2020. 1