Localized Graph-Based Neural Dynamics Models for Terrain Manipulation

Chaoqi Liu[1], Yunzhu Li[2], Kris Hauser[1]

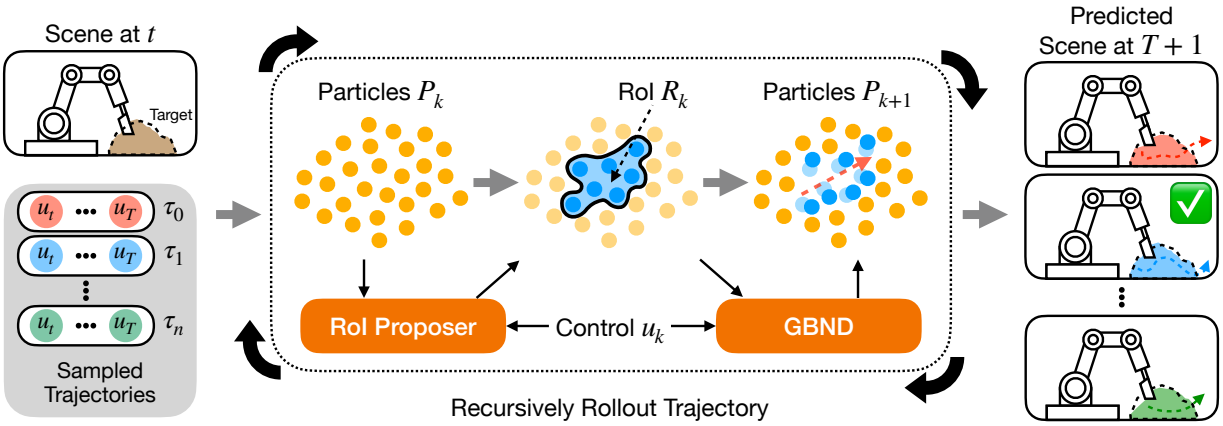


Fig. 1: Smooth, collision-free motions generated by INSATxGCS (IxG) for various tasks with three Motoman HC10DTP arms operating simultaneously in a 18-DoF multi-arm assembly scenario.

***Abstract*—Predictive models significantly enhance robot performance in terrain manipulation tasks, crucial for applications such as construction and planetary exploration. However, accurately modeling terrain deformation is computationally demanding due to the high-dimensional nature of detailed terrain representations. This paper presents a scalable learning-based framework for terrain dynamics prediction and manipulation using Graph-based Neural Dynamics (GBND). Our approach represents terrain deformation through particle graphs, dynamically identifying a small, active subgraph (hundreds of particles) within a potentially massive terrain graph (millions of particles). To efficiently determine this active region, we introduce a learned Region of Interest (RoI) proposer, driven by robot control inputs and current scene observations. Additionally, we propose novel boundary feature encodings to ensure realistic dynamics prediction within the RoI, effectively preventing particle penetration at boundaries. Experiments demonstrate that our method achieves superior prediction accuracy while significantly outperforming naive GBND approaches in computational efficiency. We validate our approach through excavation and shaping tasks across terrains of varying granularity. Project page: chaoqi-liu.com/scoopbot.**

I. INTRODUCTION

Autonomous terrain manipulation is critical for tasks in construction and planetary exploration, where human labor may be limited or unavailable [1, 2]. Robots performing these tasks must accurately predict terrain deformation resulting from interactions such as scooping, dumping, or pushing. Traditional analytical or geometric modeling techniques have limited flexibility and typically require extensive parameter tuning [3]. Terramechanics-based simulators can address more complex scenarios but are computationally intensive and challenging to calibrate [4, 5].

[1]University of Illinois at Urbana-Champaign [2]Columbia University. This work was partially funded by NSF Grant #FRR-2409661. Corresponding author: chaoqil2@illinois.edu.

Learning-based approaches provide an alternative, directly capturing complex terrain behaviors from data [6, 7]. Graph-based Neural Dynamics (GBND), which employ Graph Neural Networks (GNNs) to model particle interactions, have demonstrated success in deformable object manipulation tasks due to their strong relational inductive biases [8, 9]. Prior GBND approaches have been applied to rigid bodies [8], elastic-plastic materials [10], fluids, and granular media [9]. However, these methods have primarily focused on bounded environments and tabletop scenarios with limited volume and static observation setups.

This paper extends GBND methods to large-scale, potentially unbounded terrains represented by millions of particles – far exceeding typical GPU memory capacities. Our core innovation is a learned Region of Interest (RoI) proposer, dynamically identifying a localized subset of terrain particles likely to move during interactions. Restricting computations to this smaller active subgraph substantially reduces computational load while enhancing prediction accuracy. Additionally, we introduce novel boundary feature encodings, ensuring accurate predictions inside the RoI and preventing unrealistic particle behaviors at boundaries.

Experiments validate our approach on excavation and shaping tasks across different terrain materials, demonstrating significant improvements in computational efficiency and prediction accuracy compared to traditional simulators and geometric RoI baselines.

II. METHOD

A. Overview and Background

We aim to efficiently predict terrain deformation from robot-terrain interaction to support planning. Given terrain state x_t and robot control u_t, we predict $\hat{x}_{t+1} = f(x_t, u_t)$ at 10 Hz. We

leverage Graph-based Neural Dynamics (GBND) [9], modeling the environment as particles in a graph $x_t \equiv G_t(V_t, E_t)$, where nodes $v_t^{(i)}$ represent particles and edges $e_t^{(k)}$ denote local interactions. Dynamics are learned using Graph Neural Networks (GNNs) with message passing, which consists of three steps. First, node and edge features are encoded into latent representations:

$$z_0[v_t^{(i)}] = f_V^{\text{enc}}(v_t^{(i)}), \quad z_0[e_t^{(k)}] = f_E^{\text{enc}}(e_t^{(k)}). \tag{1}$$

Next, latent messages propagate through the graph for L iterations. At each iteration l, edge and node latent vectors are updated:

$$z_{l+1}[e_t^{(k)}] = f_E^{\text{prop}}(z_l[e_t^{(k)}], z_l^i, z_l^j), \tag{2}$$

$$z_{l+1}[v_t^{(i)}] = f_V^{\text{prop}}\Big(z_l[v_t^{(i)}], \sum_{e \in E^+(v_t^{(i)})} z_l[e]\Big). \tag{3}$$

Finally, a node decoder predicts particle displacements:

$$\Delta\hat{p}_t^{(i)} = f_V^{\text{dec}}(z_L[v_t^{(i)}]), \quad \hat{p}_{t+1}^{(i)} = p_t^{(i)} + \Delta\hat{p}_t^{(i)}. \tag{4}$$

The encoder, propagation, and decoder functions (f_V^{enc}, f_E^{enc}, f_V^{prop}, f_E^{prop}, f_V^{dec}) are trained via backpropagation.

B. GBND for Large-Scale Terrain Modeling

To model terrain dynamics, we first construct a particle-based scene graph from perception data. Given an observed terrain surface map, we densely instantiate *terrain particles* $P_t = \{p_t^{(i)}\}_{i=1}^{N_p}$ throughout the terrain volume to a predefined depth. We also instantiate *tool particles* on the robot's end effector. Particles are connected to their k nearest neighbors within a specified distance threshold.

Our baseline node features are defined as:

$$v_t^{(i)} \triangleq [\mathbf{v}_{t-H:t-1}^{(i)}, u_t^{(i)}, c^{(i)}]^\top, \tag{5}$$

where $\mathbf{v}_{t-H:t-1}^{(i)}$ represents particle velocities over the last H time steps, $u_t^{(i)}$ indicates direct robot-induced impulses, and $c^{(i)}$ encodes the particle class (terrain or tool). Edge features connecting particles i and j are defined as:

$$e_t^{(k)} \triangleq [p_{t-H:t}^{(i)} - p_{t-H:t}^{(j)}, c^{(i)}, c^{(j)}]^\top. \tag{6}$$

To improve computational efficiency, we dynamically identify a small Region-of-Interest (RoI) $R_t(u_t) \subset \mathbb{R}^3$, defined by a learned implicit function $g(p, u_t)$ (Sec. II-D). Only particles inside the RoI are considered dynamic and form the active subgraph used for GBND propagation, significantly reducing computational costs while preserving accuracy.

C. Encoding RoI Boundaries as Node Features

Standard GBND approaches typically assume fixed, bounded environments, implicitly learning boundary conditions such as floors or walls [10, 11]. However, in large-scale terrain manipulation with dynamic regions of interest (RoI), the boundary conditions are not strict boundaries but rather static terrain material unaffected by current interactions.

To accurately capture interactions at these RoI boundaries, we incorporate estimated geometric normals into node features. Particle normals are computed from local RoI point clouds using standard estimation methods [12]. Near boundaries, these normals provide valuable directional resistance cues, effectively preventing unrealistic particle penetration. Furthermore, encoding normals ensures translation invariance, i.e., the model predictions remain consistent irrespective of absolute particle positions:

$$\forall \Delta p \in \mathbb{R}^3, \quad f(p_t^{(i)} + \Delta p, u_t) = f(p_t^{(i)}, u_t) + \Delta p. \tag{7}$$

We empirically validate our boundary encoding. Compared to baseline encodings (no encoding, and absolute height 'z features'), our normal-based encoding demonstrates superior generalization, maintaining accuracy across both in-distribution and out-of-distribution test scenarios.

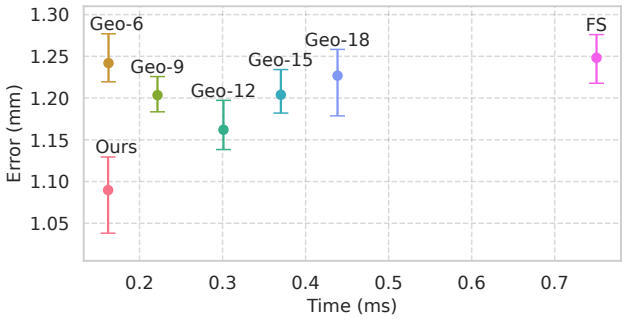


Fig. 2: Comparing our method against full-scale GBND (FS) and geometric region proposers with different size (Geo-X). Our method demonstrates significant advantages in speed and GPU memory. $\approx$ 3,000 particles and batch size 256 are used in these experiments, per sample measurements are reported (i.e., divided by 256). Colors and labels are shared by both figures.

D. Convolutional Neural Net RoI Proposer

We define the region-of-interest (RoI) as an implicit function $g(p, u_t)$, determined via a CNN-based heightmap predictor. Specifically, we represent the RoI using a tool-centered heightmap $\tilde{g}(x, y)$, where points whose vertical coordinate exceeds $\tilde{g}(x, y)$ belong to the RoI, i.e., $g(p, u_t) = p_z - \tilde{g}(T_{\text{tool}}^{-1}(p_x, p_y))$. We refer readers to our complete paper for a detailed discussion on the RoI proposer.

E. Planning Terrain Manipulation Trajectories

Our terrain manipulation planner selects among parametrized scooping, pushing, and dumping actions to minimize the predicted terrain shape difference from a target heightmap. After each executed trajectory, the terrain is updated, and planning repeats.

We adopt the widely-used penetrate-drag-scoop (PDS) trajectory parametrization [1, 6] for scooping and pushing. Dumping trajectories use a simpler 6D parameter space (location, yaw, pre- and post-dump pitch angles).

The planner utilizes Model Predictive Path Integral (MPPI) [13]. MPPI iteratively samples trajectory parameters, rolls out predictions using our learned dynamics model, evaluates their scores based on terrain shape differences, and updates its sampling distribution to focus on high-performing regions.

III. EXPERIMENTS AND RESULTS

The computational performance and accuracy of the proposed method were thoroughly evaluated and contrasted against simpler baselines. Furthermore, the capability of our method was assessed through its application to extended-range excavation tasks. All experimental evaluations were conducted using an Intel i9-13900K CPU and a single NVIDIA GeForce RTX 4080 GPU.

A. Implementation Details

Our GBND architecture uses multi-layer perceptrons (MLPs). We collected pre-training data using the SAPIEN simulator [14], as Nvidia Flex [15] exhibited unrealistic behaviors at higher particle counts. Synthetic terrains were generated via the diamond-square algorithm [16], producing approximately 3,000 small cubes per landscape, each populated with 5–10 particles. Particle sets were subsequently downsampled using farthest-point sampling [17].

We simulated roughly 1,000 random robot-terrain interaction trajectories and tracked downsampled particle motions for training. The dynamics model was trained by regressing particle trajectories using mean squared error (MSE). The CNN-based RoI proposer was trained using corresponding simulation data. Each forward dynamics prediction timestep was set to 0.1 s.

B. Computational Efficiency and Accuracy

We evaluate our method's efficiency and accuracy against two baselines: (1) a full-scale GBND using all terrain particles (FS), and (2) geometric RoIs (Geo-X), defined as cylinders of diameter X cm centered at the scoop tip. Due to GPU constraints, terrains are limited to 3,000 particles (50 cm $\times$ 50 cm $\times$ 5 cm, particle density 0.5/cm^3), much smaller than typical field scenarios.

Figure 2 shows our learned RoI method achieves computational performance similar to the smallest Geo-X variant, while significantly outperforming baselines in prediction accuracy. GBND inherently struggles with distant particle jitter; restricting dynamics predictions to a learned RoI introduces inductive biases that mitigate this issue. Small Geo-X RoIs fail to encompass all moving particles, causing high prediction errors, while large Geo-X RoIs suffer from jitter by unnecessarily including static particles.

Table I highlights optimal efficiency at batch size $b = 256$, with prediction time below 0.2 ms/sample, enabling planners to evaluate hundreds of trajectories in seconds. Compared to GPU-based simulation (SAPIEN 3.0 [14]), our method is an order of magnitude faster, supports larger batch sizes within GPU memory constraints, and can be fine-tuned for real-world accuracy.

C. Real-World Terrain Manipulation

We validated our approach in real-world experiments on two different materials: pebbles (0.8–1.0 cm grain size) and fine play sand (grain size $\ll$1 mm). Terrains were contained within a box of dimensions 0.9 m$\times$0.6 m$\times$0.2 m for repeatability, though the method generalizes naturally.

Batch	Ours	Geo-6	Geo-12	Geo-18	FS	SAPIEN
1	4.624	3.543	4.459	5.157	5.443	3.309 (CPU)
2	2.454	1.913	2.280	2.616	2.775	2.104
4	1.400	1.036	1.250	1.463	1.642	1.627
8	0.860	0.586	0.755	0.931	1.097	1.563
16	0.522	0.366	0.501	0.650	0.872	1.713
32	0.337	0.242	0.364	0.520	0.885	1.991
64	0.232	0.181	0.332	0.498	0.809	2.211
128	0.179	0.169	0.318	0.456	0.774	1.755
256	**0.162**	**0.163**	0.301	0.438	**0.750**	**OoM**
512	0.181	0.181	**0.293**	**0.431**	OoM	OoM
1024	0.198	0.198	0.294	OoM	OoM	OoM

TABLE I: Average prediction time (ms) per sample as the batch size varies, showing that 256 is an optimal batch size for our framework. 3000 tiny cubes are simulated in each GPU parallelized SAPIEN scenes.

To fine-tune our model to real-world data, we collected 100 robot-terrain interaction trajectories recorded by an overhead RGB-D camera. Scene point clouds were extracted from depth images, excluding robot particles via proprioception and retaining tool particles by re-sampling within the tool's convex hull. Due to inability to track individual particle identities across frames, we trained with chamfer distance [18] and earth-mover's distance [19]. This fine-tuning improved model accuracy by approximately 5–10% compared to the simulation-trained baseline.

We tested two terrain shaping tasks: (1) excavating a hole, and (2) forming a rectangular moat. Desired terrain shapes were specified as 2D heightmaps, with L_1 heightmap differences as the evaluation metric. These shapes featured sharp discontinuities, unattainable exactly due to the materials' settling behaviors (angle of repose approximately 25–40°). The planner evaluated 1,500 candidate trajectories with an average planning horizon of 30 timesteps (3 seconds), requiring 20–30 seconds total planning time. Qualitative results (Fig. 3) demonstrate successful terrain shaping.

We also implemented a heuristic baseline inspired by Yang et al. [20], which uses dynamic programming to find axis-aligned trajectories maximizing scooped material volume. Although efficient, this baseline initially makes rapid progress but fails to capture detailed terrain shapes due to neglecting granular material settling (Fig. 4).

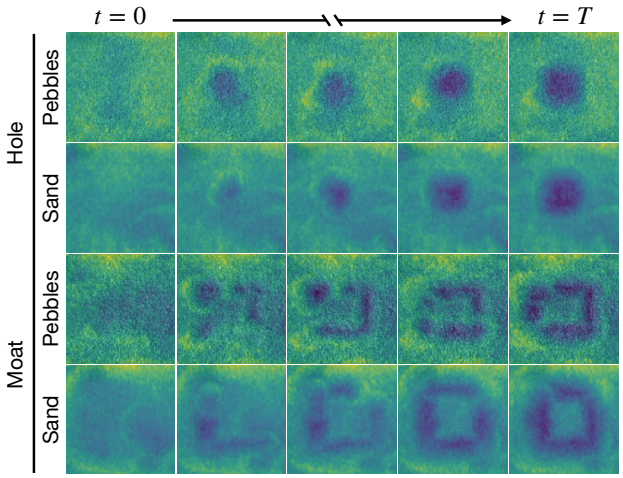
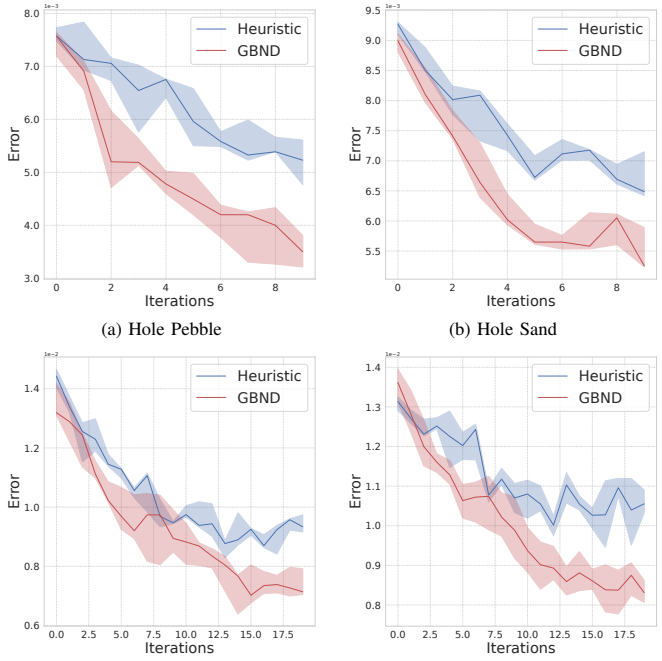


Fig. 3: Sequences of heightmaps recorded during real-world manipulation.



(a) Hole Pebble (b) Hole Sand

(c) Moat Pebble (d) Moat Sand

Fig. 4: Errors between the observed terrain and the target terrain after each PDS trajectory. Interquartile ranges are shaded. 10 trials per scenario.

IV. CONCLUSION AND LIMITATION

We presented a framework for handling unbounded terrains in graph-based neural dynamics (GBND) models that simultaneously learns a RoI and particle dynamics. Novel node features are key to success of this model. Experiments show that this method achieves orders of magnitude faster prediction and GPU memory usage compared with naïve GBND models or GPU-based physics simulators. It can be trained with both simulated and real data, and when used as a model for planning our method can achieve desired target shapes on real-world excavation platforms.

There remains some limitations of our work that we would like to address in future work. First, our RoI proposer does not accurately model long-range terrain effects (e.g., landslides). Second, our current platform uses an overhead camera to update the terrain, and we would like to extend this to egocentric 3D mapping, in which our model may need to model occluded parts of the terrain. Third, our models assume terrain homogeneity, and future work should address heterogeneous materials including terrains with embedded obstacles. Lastly, the current dynamics model requires full-parameter fine-tuning when moving to unseen materials, one potential solution is to adapt physics-parameter conditioned GBND [21] for online adaptation.

REFERENCES

[1] Yifan Zhu, Pranay Thangeda, Melkior Ornik, and Kris Hauser. Few-shot adaptation for manipulating granular materials under domain shift, 2023.

[2] Ryan Lee, Benjamin Younes, Alexander Pletta, John Harrington, Russell Q. Wong, and William "Red" Whittaker. Cratergrader: Autonomous robotic terrain manipulation for lunar site preparation and earthmoving, 2024.

[3] A. R. Reece. Paper 2: The fundamental equation of earth-moving mechanics. *Proceedings of the Institution of Mechanical Engineers, Conference Proceedings*, 179(6):16–22, 1964.

[4] H.J. Herrmann. Simulation of granular media. *Physica A: Statistical Mechanics and its Applications*, 191(1):263–276, 1992.

[5] Nathan Bell, Yizhou Yu, and Peter J. Mucha. Particle-based simulation of granular materials. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '05, page 77–86, New York, NY, USA, 2005. Association for Computing Machinery.

[6] Sanjiv Sing. *Synthesis of tactical plans for robotic excavation*. PhD thesis, Citeseer, 1995.

[7] H. J. Terry Suh and Russ Tedrake. The surprising effectiveness of linear models for visual foresight in object pile manipulation, 2020.

[8] Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. *arXiv preprint arXiv:1810.01566*, 2018.

[9] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks, 2020.

[10] Haochen Shi, Huazhe Xu, Zhiao Huang, Yunzhu Li, and Jiajun Wu. Robocraft: Learning to see, simulate, and shape elasto-plastic objects with graph networks, 2022.

[11] Haochen Shi, Huazhe Xu, Samuel Clarke, Yunzhu Li, and Jiajun Wu. Robocook: Long-horizon elasto-plastic object manipulation with diverse tools. *arXiv preprint arXiv:2306.14447*, 2023.

[12] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique signatures of histograms for local surface description. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 356–369, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[13] Grady Williams, Paul Drews, Brian Goldfain, James M. Rehg, and Evangelos A. Theodorou. Aggressive driving with model predictive path integral control. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1433–1440, 2016.

[14] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. SAPIEN: A simulated part-based interactive environment. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[15] Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. Unified particle physics for real-time applications. *ACM Transactions on Graphics (TOG)*, 33(4):1–12, 2014.

[16] Alain Fournier, Don Fussell, and Loren Carpenter. *Computer rendering of stochastic models*, page 189–202. Association for Computing Machinery, New York, NY, USA, 1998.

[17] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, 2017.

[18] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017.

[19] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40:99–121, 2000.

[20] Yajue Yang, Pinxin Long, Xibin Song, Jia Pan, and Liangjun Zhang. Optimization-based framework for excavation trajectory generation. *IEEE Robotics and Automation Letters*, 6(2):1479––1486, 2021.

[21] Kaifeng Zhang, Baoyu Li, Kris Hauser, and Yunzhu Li. Adaptigraph: Material-adaptive graph-based neural dynamics for robotic manipulation, 2024.