Supplemental Material
TNS: Terrain Traversability Mapping and Navigation System for Autonomous Excavators

Tianrui Guan[1]* Zhenpeng He[1] Ruitao Song[1] Dinesh Manocha[2] Liangjun Zhang[1]
[1] Robotics and Auto-Driving Laboratory, Baidu Research
[2] University of Maryland, College Park

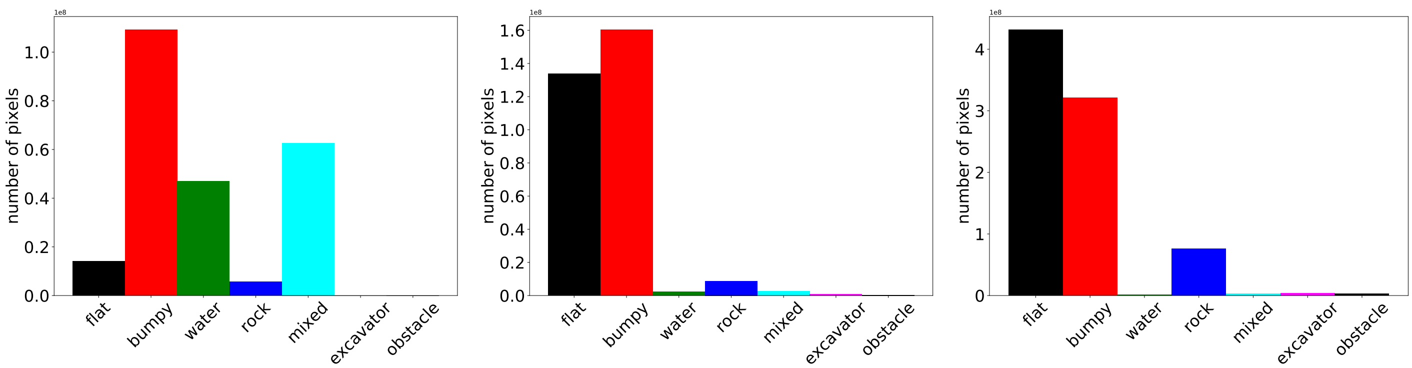

Fig. 1: **Label distribution of each image sequence.**

I. More Details of the CWT dataset

Our dataset is collected at a construction site while an excavator is navigating through the work area. We collect 3 videos that total approximately 30 minutes; 669 images of size 1920×1080 with pixel-wise annotation are included in our dataset. Please refer to this link for the access to the CWT dataset.

A. Dataset Details and Statistics

There are three video sequences collected on our excavator test site. In Figure 1, we show the class distribution breakdown for three sequences. The first video is collected after rain and consists of mostly water and muddy ground. The trenches caused by excavation and navigation can also be seen. The other two video sequences are captured on a sunny day in different scenarios. The videos are collected by a professional operator controlling the movement of the robot. The three videos are 268s, 668s, and 822s. We sample the camera stream every two seconds and annotate the images with ground truth labels, resulting in a total of 669 images after removing some redundant ones.

B. Benchmarks

We give several metrics and show the performance of several SOTA methods on the CWT dataset in Table I. C denotes the set of all classes.

* Work done during an internship at Baidu RAL.

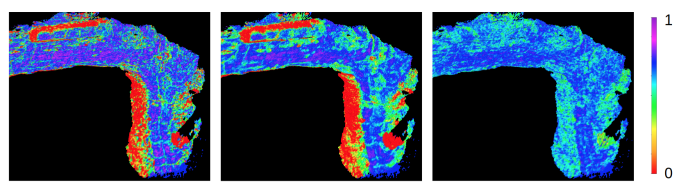

Fig. 3: **Grid map output comparison:** We show slope **(left)**, step height **(middle)**, and roughness **(right)** values in geometric traversability computations. All values are converted to the scale from 0 to 1. Slope value tends to have many small areas of peaks, while step height tends to have smoother values across a bigger region. On the other hand, roughness does not have many peak values, and many regions like hills are already captured by the previous two measurements.

$$mIoU = 1/c \sum_{c=1}^{C} \frac{TP_c}{TP_c + FP_c + FN_c}$$

$$mAcc = 1/c \sum_{c=1}^{C} \frac{TP_c}{TP_c + FN_c}$$

$$aAcc = \frac{\sum_{c=1}^{C} TP_c}{Numbers\ of\ All\ Pixels}$$

Year	Methods	Flat	Bumpy	Water	Rock	Mixed	Excavator	Obstacle	mIoU	mAcc
2018	CGNet [10]	73.02	63.11	38.22	69.67	47.0	47.04	35.78	53.41	67.59
2019	Fast SCNN [6]	74.1	65.87	32.02	73.42	46.58	45.51	45.91	54.77	68.75
2019	Fast FCN [9]	71.96	61.23	35.61	60.06	35.3	0.0	27.6	41.68	51.85
2021	BiSeNetV2 [12]	76.49	69.65	38.33	71.44	46.42	41.02	37.22	54.37	67.05
2021	SETR* [13]	54.24	49.67	4.07	25.23	6.03	0.0	0.16	19.91	30.61
2021	DPT* [7]	59.45	53.75	23.78	33.69	26.0	0.0	6.49	29.02	47.65
2021	Segformer [11]	73.44	64.47	39.62	70.29	43.81	30.48	32.07	50.6	64.29

TABLE I: **Performance of SOTA methods on the CWT dataset:** We list several SOTA semantic segmentation methods and train the model with 240K iterations. $*$ marks methods that do not converge well after 240K additional iterations.

II. More Details of TNS

A. Roughness in Geometric Traversability

In many existing works [1, 8] for geometric traversability or danger value calculation, a roughness score is calculated as a factor of terrain traversability.

Roughness Estimation: The terrain roughness r is calculated as the standard deviation of the terrain height values to the fitting plane. The distance d from the center point p of the grid to the fitting plane of k neighboring grids is calculated as:

$$d = \frac{\vec{p\bar{p}} \cdot \vec{n}}{|\vec{n}|}$$

where $\vec{n}$ is the surface normal vector of the fitting plane and $\bar{p}$ is a point in the plane. Finally, the roughness estimation of the grid g can be computed as:

$$r = \sqrt{\sum_{i=1}^{k} (d_i)^2}$$

However, roughness is not a good measurement in unstructured environments, especially in our situation. During the design of our method, we discover that the roughness measurement is either random or, in some regions, the distribution of roughness resembles that of slope or step height, except with lower peak values. Eventually, the roughness score did not impact the results too much. We demonstrate such similarity in Figure 3.

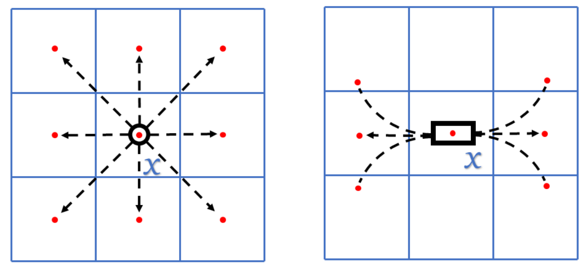

Fig. 4: **Left:** A* associates costs with centers of cells and only visits states that correspond to grid-cell centers. **Right:** Hybrid A* associates a continuous state with each cell, and the score of the cell is the cost of its associated continuous state. A* path can always move to the center of the adjacent node, while in hybrid A*, we consider the actual motion constraints of the object, so the red dot does not appear in the grid center.

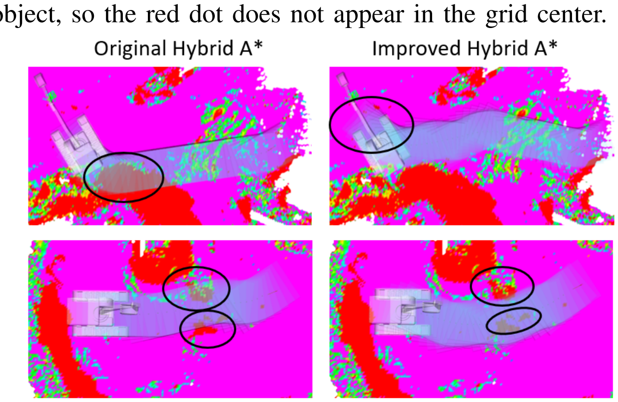

Fig. 5: **Comparison between original and improved hybrid A* planners.** Our planning method has more flexibility, including running over small obstacles between two tracks based on our traversability map.

III. More Details of TNS-based Planning

In this section, we add more details of our planning method and experimentation.

A. A* and Hybrid A*

A* [3] search can be seen as an improvement of Dijkstra's search. Dijkstra calculates the cost to start $g(x)$ of each vertex to determine the next vertex to be expanded. A* search enhances the algorithm by using heuristic cost $h(x)$, allowing faster convergence under certain conditions, while still ensuring its optimality [4]. The heuristic cost $h(x)$ is the cost to goal based on a heuristic estimate of the cost from state x to the goal state x_{goal}, since the actual cost $g(x)$ is the path that has been actually traversed. The total cost is thus

$$f(x) = g(x) + h(x)$$

by which the way points will be sorted. A standard heuristic estimate function is the Euclidean distance for two dimensional problems.

The hybrid A*[2, 5] algorithm is proposed for path planning of nonholonomic robots. In A*, we do not consider the direction of the moving object, and we do not consider the actual movement of the object. However, in hybrid A*, we need to consider the constraint of the robot motion model. In Figure 4, we use the red dot to indicate the possible position of the robot. The differences between the two algorithms are shown in Table II, and we also provide pseudo-code in Alg. 1.

	Hybrid A*	A*
Dimension	(x, y, θ)	(x, y)
Vertex	Possible movement paths	Grid map cells
g(x)	Kinematic model	Manhattan / Euclidean
h(x)	Max(Reeds_Shepp Dist, A*)	Manhattan / Euclidean

TABLE II: Differences between A* and Hybrid A*

Algorithm 1: Hybrid A* Search

Input: Start state: x_s; Goal state: x_g
Output: Valid path between x_s and x_g

begin
  $O = \emptyset$ // Initialize Open set
  $C = \emptyset$ // Initialize Close set
  $f(x_s) = g(x_s) + h(x)$ // Update cost of x_s according to the cost function. $g(x)$ is the actual cost and $h(x)$ is the heuristic cost.
  $O.push(x_s)$
  while O *not empty* **do**
    $x \leftarrow O.popMin()$ // $O.popMin()$ return the node with the lowest cost in O
    if $x == x_g$ **then**
      return *path* // Trace the parent node from the end point x, until it reaches the starting point, return to the result *path* found.
    else
      $C.push(x)$
      for *each* $n \in neig(x)$ **do**
        // Go through all collision-free neighbors of x according to the kinematic model
        if $n \notin O$ **then**
          $f(n) = n.updateCost()$
          $O.push(n)$
  return *null* // Can not find a valid path

Scenarios	Difficult Terrain	Obstacles	Geometric Method [1] (%)	TNS (%)
Case 1	✓	✓	40	**60**
Case 2	✓	✓	16.67	**71.43**
Case 3	✓		50	50
Case 4	✓	✓	50	**100**
Case 5		✓	20	**100**
Case 6	✓	✓	40	**80**
Case 7		✓	50	50
Case 8		✓	20	**60**
Case 9	✓		20	**60**
Overall	✓	✓	33.3	**82.6**

TABLE III: **Success rate of planning in each scenario.** "Difficult Terrain" means the excavator must traverse through or navigate around a rough region or water. "Obstacles" means there are obstacles in the environment.

B. Experiment Details of Offline Planning

We compare the success rate of the planner using output traversability maps from the geometric-only method [1] and the proposed TNS. We use 9 different scenarios, and each scenario is tested with the same starting points and random goal position with more than 10 trials. We list the details of all scenarios in Table III.

C. Failure Cases of Hybrid A* in Our Applications

We show some planning scenarios where traditional Hybrid A* would fail in Fig 5.

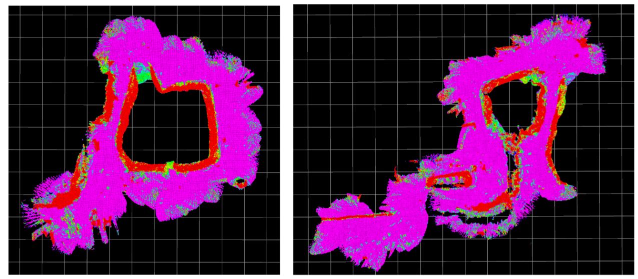

Fig. 6: **Traversability maps of our two testing sites.** Each grid is 10 m by 10 m.

IV. More Visualization

A. Testing Site

We show the traversability maps of two testing sites in Figure 6. In Figure 7, we show a drone image taken in 2020. Note that the image is outdated, and the condition might be different from when our experiments are done.

B. Qualitative Comparisons on Mapping methods

In Figure 8, we compare traversability maps generated using a geometric-only method [1] and using TNS with geometric-semantic fusion. The output after fusion is less noisy since segmentation results can smooth out safe regions. Our method detects more non-traversable regions based on obstacles and dangerous regions from semantic information.

Acknowledgement

This work was done as as summer intern at Baidu RAL. We appreciate the discussion and support from Baidu RAL team.

Fig. 7: **Aerial image of our testing site taken in 2020.**

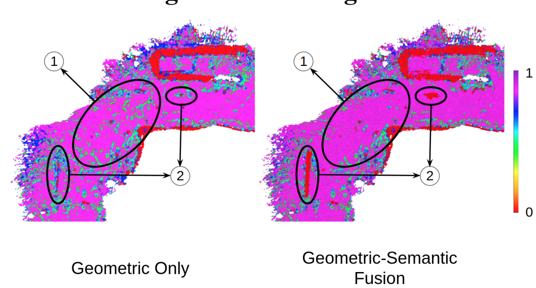

Fig. 8: **Grid map comparison between the geometric-only scheme [1] and ours:** (1) Our method is less noisy and has more connected regions to plan a feasible trajectory. (2) Our method can detect obstacles that the geometric method could not recognize.

References

[1] Annett Chilian and Heiko Hirschmüller. Stereo camera based navigation of mobile robots on rough terrain. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4571–4576, 2009. doi: 10.1109/IROS.2009.5354535.

[2] Dmitri Dolgov, Sebastian Thrun, Michael Montemerlo, and James Diebel. Practical search techniques in path planning for autonomous driving. *Ann Arbor*, 1001 (48105):18–80, 2008.

[3] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968. doi: 10.1109/TSSC.1968.300136.

[4] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[5] Karl Kurzer. Path planning in unstructured environments : A real-time hybrid a* implementation for fast and deterministic path generation for the kth research concept vehicle. Master's thesis, 2016.

[6] Rudra P. K. Poudel, Stephan Liwicki, and R. Cipolla. Fast-scnn: Fast semantic segmentation network. In *BMVC*, 2019.

[7] Rene Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, 2021.

[8] Martin Wermelinger, Péter Fankhauser, Remo Diethelm, Philipp Krüsi, Roland Siegwart, and Marco Hutter. Navigation planning for legged robots in challenging terrain. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1184–1189, 2016. doi: 10.1109/IROS.2016.7759199.

[9] Huikai Wu, Junge Zhang, Kaiqi Huang, Kongming Liang, and Yu Yizhou. Fastfcn: Rethinking dilated convolution in the backbone for semantic segmentation, 2019.

[10] Tianyi Wu, Sheng Tang, Rui Zhang, and Yongdong Zhang. Cgnet: A light-weight context guided network for semantic segmentation. *IEEE Transactions on Image Processing*, 30:1169–1179, 2021.

[11] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=OG18MI5TRL.

[12] Changqian Yu, Changxin Gao, Jingbo Wang, Gang Yu, Chunhua Shen, and Nong Sang. Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation. *International Journal of Computer Vision*, 129:1–18, 11 2021. doi: 10.1007/s11263-021-01515-2.

[13] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H.S. Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, 2021.