Monocular Simultaneous Localization and Mapping using Ground Textures

Kyle M. Hart[1,2], Brendan Englot[2], Ryan P. O'Shea[1], John D. Kelly[1], David Martinez[1]

***Abstract*— Recent work has shown impressive localization performance using only images of ground textures taken with a downward facing monocular camera. This provides a reliable navigation method that is robust to feature sparse environments and challenging lighting conditions. However, these localization methods require an existing map for comparison. Our work aims to relax the need for a map by introducing a full simultaneous localization and mapping (SLAM) system. By not requiring an existing map, setup times are minimized and the system is more robust to changing environments. This SLAM system uses a combination of several techniques to accomplish this. Image keypoints are identified and projected into the ground plane. These keypoints, visual bags of words, and several threshold parameters are then used to identify overlapping images and revisited areas. The system then uses robust M-estimators to estimate the transform between robot poses with overlapping images and revisited areas. These optimized estimates make up the map used for navigation. We show, through experimental data, that this system performs reliably on many ground textures, but not all.**

I. Introduction

When exploring unknown regions, robotic ground systems frequently rely on Simultaneous Localization and Mapping (SLAM) to map their surroundings and track their positions through the environment. While a wide range of sensors can be used, from lidar to vision, monocular cameras are a popular choice due to their low cost and rich information content.

These monocular SLAM systems frequently look out into the world for salient features to use for navigation. However, some environments, such as flat open spaces, lack enough features to reliably navigate. Additionally, cameras are sensitive to illumination changes in the environment, such as glare from a setting sun.

In these scenarios, the only consistent source of features comes from the surface the ground robot is traveling on. Recent work has shown that, despite its unstructured appearance, some ground textures are sufficiently distinct enough to successfully support localization [1], [2], [3]. These methods provide a reliable source of information, even in flat, open spaces or other feature sparse environments. Additionally, a downward facing camera is much more robust to illumination changes, since it has a limited, shielded field of view.

[1]Naval Air Warfare Center, Aircraft Division (NAWCAD), Lakehurst, NJ, 08733, USA, kyle.m.hart2.civ@us.navy.mil

[2]Department of Mechanical Engineering, Stevens Institute of Technology, Hoboken, NJ, 07030, USA, benglot@stevens.edu

Distribution Statement A: Approved for public release; distribution is unlimited, as submitted under NAVAIR Public Release Authorization 2022-0586. The views expressed here are those of the authors and do not reflect the offical policy or position of the U.S. Navy, Deparment of Defense, or U.S Government.

Here, we consider expanding upon ground texture localization methods by removing the requirement for an *a priori* ground texture map. Without this need, an operator can save time on initial setup, since a complete map does not need to be created prior to operation. This allows exploration in previously unknown environments. Additionally, the system becomes more robust to changes in the environment that would differ from an *a priori* map. This capability is accomplished through the introduction of a full SLAM system. In particular, our contributions in this systems paper are as follows:

- To the best of our knowledge, development of the first online ground texture SLAM system using only a monocular camera.
- A unique algorithm within the ground texture domain that exploits the known depth of ground texture images when estimating the transform between overlapping images and identifying loop closures.
- Experimental results on a recent data set showing centimeter level accuracy on some textures and superior performance across changing textures, as well as consistent, accurate loop closure identification.

First, we will discuss some related works in Section II and formulate the target problem in Section III. Then, Section IV will detail our approach to the target problem. Lastly, Section V will highlight supporting experimental results. The source code for this system is available at **https://github.com/Navy-RISE-Lab/ground-texture-slam**.

II. Related Work

Numerous SLAM systems incorporate monocular camera information. Some couple it with additional sensor information, such as inertial measurement units or lidar sensors [4], [5]. Others rely on just the camera as the only sensor. Our work aligns with this second class of systems.

Monocular camera only SLAM systems can be grouped into multiple categories distinguished both by the method used to compare images and by how the maps are stored. Image comparison generally falls into *direct methods* or *indirect methods*. *Direct methods* use the pixel intensities to compare images and include systems such as [6]. *Indirect methods* use keypoints, which are salient points in the image identified through a variety of different algorithms, such as ORB and SIFT [7], [8]. Map storage typically includes *dense* or *sparse* maps. Dense maps store the entire image in the map, as in [9]. Sparse maps store a subset of information, such as keypoint values, as in [6]. The work described in this paper falls in the *indirect-sparse* category.

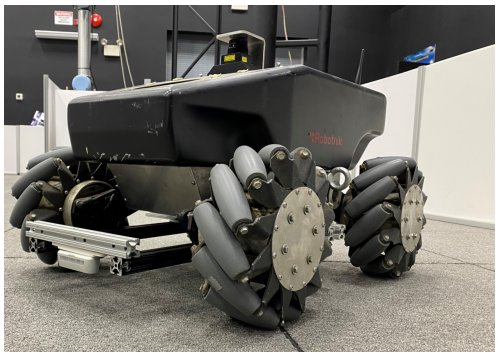


Fig. 1: A downward facing camera setup configured by the authors. Note this is not the one used for gathering the experimental data considered in this paper, but is illustrative of a typical setup.

Other *indirect-sparse* monocular SLAM systems include the popular ORB-SLAM series [10], [11], [12]. In these systems, keypoints are identified in images and their real-world locations are estimated to use for later localization and loop closure. Similar algorithms exist that look for domain-specific features, such as points and lines [13], [14].

In many monocular SLAM systems, a major component of the algorithm attempts to perform accurate depth estimation. While images are information rich, the 2D nature loses depth information. To account for this, monocular SLAM systems frequently use either traditional methods for depth estimation [15], or use machine learning [16], [17].

For systems that use the ground texture for features, the depth estimation problem vanishes as all features are at the same depth. However, the current state of the art is limited to localization with an *a priori* map of the environment. With these approaches, multiple images are taken of the ground such that the entire operating area can be found in at least one image. Then, each algorithm searches the map to find the closest matching images.

Most of these ground texture localization methods use keypoints and descriptors as in other monocular approaches [1]. Micro GPS uses a voting scheme where each keypoint votes for a closely matched image, then uses RANSAC to estimate a transform between images [18]. [19] uses a similar keypoint approach, but explores keypoint determination via both a uniform distribution across the image and random sampling without regard to image content. Lastly, [20] uses a neural network to identify keypoints in images.

III. Problem Description

Here, we consider a ground robot equipped only with a downward facing, calibrated, monocular camera with intrinsic matrix $\mathbf{K} \in \mathbb{R}^{3\times3}$ and known 3D pose relative to the robot's origin on the ground plane. This pose can be represented as the homogeneous matrix $\mathbf{T}_{RC} \in \mathbb{R}^{4\times4}$, which is used to transform data measured in the camera's frame of reference, C, into the robot's frame of reference, R. Fig. 1 shows an example setup.

The robot travels over a planar ground surface through several poses, $\vec{x_t}$. These poses are defined in the ground plane as the robot's 2D pose as measured from the world or map frame, W:

$$\vec{x_t} = \begin{pmatrix} x_t & y_t & \theta_t \end{pmatrix}^\top \tag{1}$$

and can be represented by the homogeneous transform $\mathbf{T}_{W\vec{x_t}} \in \mathbb{R}^{3\times3}$. At each of these poses, the robot receives an observation, $\mathbf{Z}_t$, in the form of a distortion free image of the ground texture.

The goal is to develop an algorithm that can reliably estimate the robot's poses, $\vec{x_t}$, for all t, using only the observations, $\mathbf{Z}_{0:t}$, the camera calibration matrix, $\mathbf{K}$, and the pose of the camera with respect to the robot, $\mathbf{T}_{RC}$.

Note that while odometry and inertial information are often available, this system specifically explores the ability to accomplish this goal without extra sensor information. Accomplishing this goal requires an approach that can estimate relative motion between successive images, and accurately detect previous sections of the terrain that have been revisited.

IV. Proposed Approach

We propose an algorithm that performs SLAM in three steps. Fig. 2 shows an outline of the proposed approach. First, incoming images are processed. Then, successive pairs of images are used to estimate visual-only odometry. Then, loop closures are exploited to correct drift. Both the odometry and loop closure steps use identified keypoints from the image along with their associated descriptors. Additionally, both steps estimate the transform between pairs of images using keypoints projected into the ground plane and M-estimators, which are robust deterministic models. Unlike the local visual odometry, loop closure detection uses three metrics to determine if a candidate loop closure is valid. The transforms estimated from each step are inserted into a factor graph that represents the map. The overall process of the algorithm is described below.

A. Image Processing

When an image is received, it is first processed to extract keypoints and their associated descriptions using the ORB algorithm [7]. Then, the keypoints are converted from pixel points to ground points. By performing this conversion, the system can directly estimate the robot's transform in real-space instead of estimating the essential matrix and converting. In most 2D to 3D projection scenarios, this conversion is only to a scale factor of the depth. However, because the distance between the camera and ground plane is known, these points can be unambiguously projected. The points are first projected from pixel values to meters, as measured from the camera's frame of reference using the following equation:

$$\mathbf{z_C} = \mathbf{K}^{-1} \cdot \mathbf{z_I} \times d. \tag{2}$$

In this equation d is the distance between the camera and the ground plane, as derived from $\mathbf{T}_{RC}$, and $\mathbf{z_I} \in \mathbb{R}^{3xN}$ is the collection of keypoints, in pixels, for this image, represented

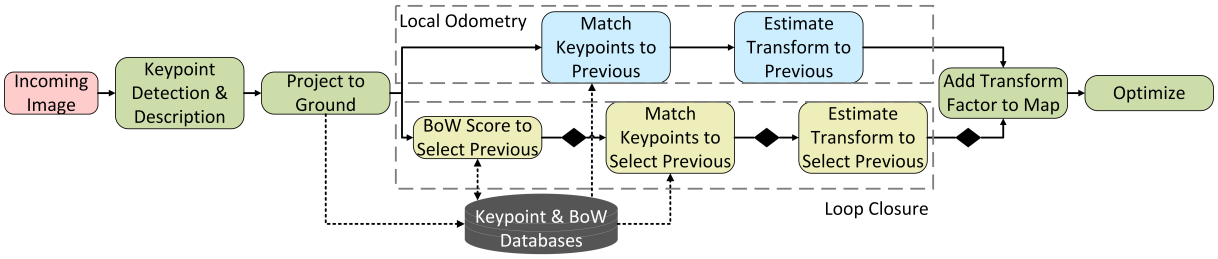


Fig. 2: System structure for a single received image. There are two substructures: local odometry (blue boxes) that compares to the previous image; loop closure (yellow boxes) that compares to almost all previous images, with thresholding (indicated by the black diamonds). The remaining steps are either incoming data (red), general processing steps (green), or information stored between iterations (gray).

in homogeneous coordinates as follows:

$$\mathbf{z_I} = \begin{bmatrix} \vec{z}_I^{(0)} & \vec{z}_I^{(1)} & \dots & \vec{z}_I^{(N)} \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & \dots & x_N \\ y_1 & y_2 & \dots & y_N \\ 1 & 1 & \dots & 1 \end{bmatrix}. \tag{3}$$

The result is a collection of 3D vectors representing the point as measured from the camera's frame of reference, in meters (with some abuse of notation turning the homogeneous representation's 1 into the Z-component). This value can then be projected into the robot's frame of reference through conversion to a 3D homogeneous representation and a simple transformation:

$$\mathbf{z_R} = \mathbf{T}_{RC} \cdot \mathbf{z_C}. \tag{4}$$

From here, the Z components are dropped since the points and robot's pose are all equiplanar on the ground plane with a Z value of 0. This results in $\mathbf{z_R}$ as a collection of 2D points. After projection, the original pixel-valued keypoints are not saved. The projected keypoints and descriptors are retained for later use.

B. Local Odometry

To conduct local odometry, projected keypoints are matched to projected keypoints from the previous image. Then a transform is estimated between the images.

1) Keypoint Matching: Keypoint matching identifies corresponding keypoints that appear in two images using their descriptors. Our implementation uses the Fast Library for Approximate Nearest Neighbors (FLANN) method [21]. For each keypoint in the current image, a FLANN-based matching algorithm uses the keypoint descriptors to find two keypoints in the previous image that have the most similar descriptors. We use OpenCV's implementation [22]. This similarity is measured with a distance score, where lower scores indicate more similarity. If the two scores differ by a certain percentage, then the keypoints are considered to be a match. In other words, the keypoints match if they are significantly more similar than the next closest match. This ratio test was introduced in [8], and is applied as follows:

$$\text{match if } f_0 <= \lambda \cdot f_1 \tag{5}$$

In the above inequality, f_0 and f_1 are the closest and second-closest distance scores, respectively, and λ is the match threshold, which is often set between 0.5 and 0.7 in our method and 0.7 in the literature.

2) Transform Estimation: Matched projected keypoints are then used in an M-Estimator factor graph to estimate the transform between them, using GTSAM's expression graph feature [23]. The experiments described here use Huber, but others are available. This factor graph estimates the X, Y, and yaw components of the transform, $\mathbf{T}_{\vec{x_j}\vec{x_i}}$, which represents the transform between two of the robot's poses. The estimated transform is the one that best fits the below equation. Since the keypoints are projected onto the ground plane, as described in Section IV-A, this estimation occurs entirely in 2D real-space, offering increased efficiency over traditional 3D SLAM methods.

$$\mathbf{z}_{R_j} = \mathbf{T}_{\vec{x_j}\vec{x_i}} \cdot \mathbf{z}_{R_i} \tag{6}$$

Once estimated, the transform and associated covariance are added to the robot's SLAM factor graph as a factor between the current pose and previous pose. The system then proceeds to the loop closure identification step.

C. Loop Closures

To correct for drift, the system must correctly identify previously visited ground textures. The observations at all previously visited poses are possible candidates. Three threshold criteria are used: visual bag of words scores, the number of keypoint matches, and a covariance parameter.

1) Visual Bag of Words: The first threshold parameter is a Visual Bag of Words score. Using the technique and library described in [24], a database of previous image descriptors is built as new observations are received. To prevent redundant loop closures on adjacent observations, descriptors are not added to the database until a sufficient number of subsequent observations has been added. In other words, if the current observation just received is $\mathbf{Z_n}$, the descriptors from $\mathbf{Z_{n-k}}$ are added to the database.

The database is then queried with the current observation's descriptors to find matches. With the settings used in this work, returned scores from each previous observation in the database range from 0 to 1, with 1 being a perfect match. All results with a score above a certain threshold are considered candidate loop closures.

The vocabulary tree used to aid in descriptor matching in this step is assembled from the descriptors from a sample of images taken across all textures. As described in ORB-SLAM, this tree is general enough to work successfully on each sequence [10].

2) Number of Keypoint Matches: Then, keypoint matching is performed, as in Section IV-B.1. Any candidate loop closures with a number of matched keypoints less than the threshold are discarded.

3) Covariance Parameter: After discarding loop closure candidates that do not meet the previous threshold requirements, the remaining candidate loop closures have their transforms estimated as in Section IV-B.2. This procedure returns the estimated transform and a covariance matrix. The last threshold value is based on the covariance and is computed as the measure of maximum uncertainty of the estimate. The corresponding equation is as follows:

$$\text{score} = \log_{10}(\max(\text{eigenvalues}(\Sigma))). \tag{7}$$

In this equation, $\Sigma \in \mathbb{R}^{3\times3}$ is the covariance returned by the transform estimator.

This equation follows from the properties of eigenvalues as measures of magnitude along principal axes, therefore the maximum eigenvalue correlates to the maximum uncertainty. The logarithm provides a monotonic scaling factor to make tuning easier. Any potential loop closure greater than the threshold value is discarded.

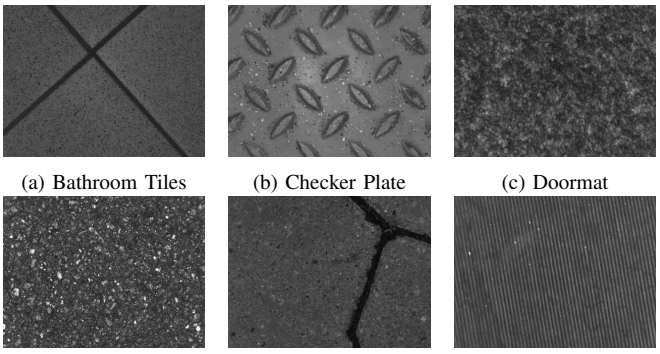
To be considered a valid loop closure, a given candidate must meet all three threshold criteria. Notably, each criterion is checked as early in the loop closure algorithm as possible. This avoids the need for costly operations when available information could discard a candidate.

If a candidate loop closure meets all three criteria, it is then added to the SLAM factor graph as an additional factor between the two poses. The graph can then be optimized using the Levenberg-Marquardt algorithm in GTSAM and the system proceeds to the next image [23]. This routine repeats throughout every image received during the robot's operation.

V. Experiments and Results

To validate this approach, results are conducted on the HD Ground Texture dataset [25]. This dataset comprises multiple different environments with multiple paths through each environment captured with a downward facing camera setup as described in Section III. The dataset also includes the ground truth poses at each image. Example images from the dataset are shown in Fig. 3. This dataset only includes approximately planar surfaces, so the ability of this system to work on non-planar outdoor surfaces, like hills, remains untested.

For testing, each image is loaded from file, then input into the SLAM system. After all images are input, the final estimated pose at each image is compared to ground truth. For comparison, Micro GPS is used to estimate poses for the same sequences [18]. Micro GPS is a state-of-the-art localization system that estimates poses by comparison to a known map. Its default parameters are used for every texture. While it is expected that a localization system will outperform a SLAM system, this comparison establishes a baseline. The mean absolute translational error for multiple paths is shown in Fig. 4, which is grouped by the type of



(a) Bathroom Tiles (b) Checker Plate (c) Doormat

(d) Footpath Asphalt (e) Parking Place (f) Ramp Rubber

Fig. 3: Example textures from the dataset, licensed under CC BY-SA 4.0 [25]. The dataset contains multiple texture environments. Each texture contains multiple sequences of observations. Each observation consists of an undistorted image and the associated ground truth at the time the image was captured.

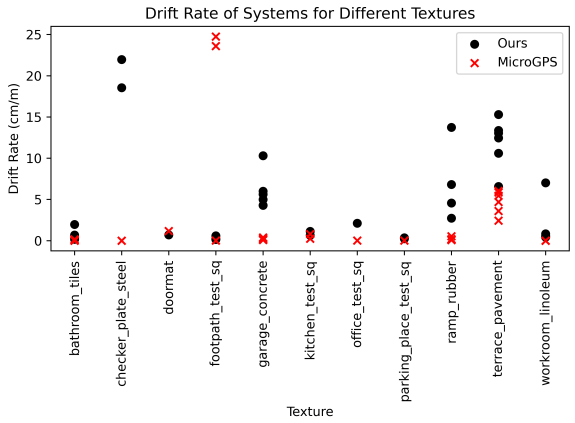


Fig. 4: The translational mean absolute error for different paths and textures. Results are normalized by the total path length due to varying length. While Micro GPS often outperforms our system, it is a localization-only approach that needs an *a priori* map.

ground texture. Due to varying path lengths, accuracy is normalized by total path length. Additionally, Fig. 5 shows the results of one path over the one texture. This figure shows both our full SLAM solution and a variant of our solution without any loop closure, to indicate the effectiveness of the loop closure at correcting drift.

Our SLAM system shows reliable performance across six of the textures, with all reported error rates under 5 cm per meter traveled. Other textures show almost uniformly poor performance, indicating that characteristics of these textures make it difficult for this system to accurately perform. Textures with poor performance are typically caused by difficulty in matching successive images during the local odometry process described in Section IV-B due to insufficient keypoint matches or inclusion of outliers. The loop closure stage is then unable to correct these errors[1].

Regardless of overall system accuracy, the system shows good loop closure identification success rates, through use of the three threshold parameters. Fig. 6 plots threshold values

[1]More detailed accuracy and loop closure results available in supplementary video and https://youtu.be/lJvTLQapsrQ

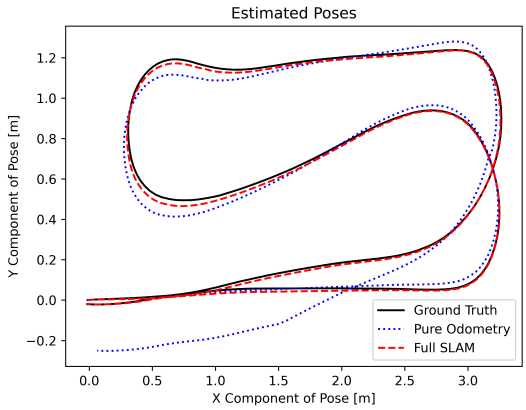


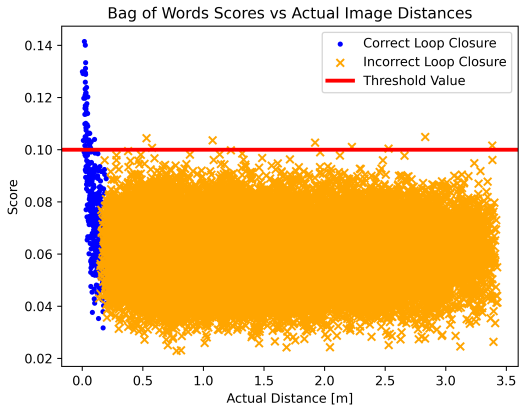
Fig. 5: Example results for the *Bathroom Tiles/test_path1* texture and sequence. The red dashed line indicates the results of our SLAM system as described above. The blue dotted line indicates our system modified to perform without any loop closure corrections.

for candidate loop closures for one of the ground textures in the HD Ground dataset [25]. It also shows the actual real world distance between these images and identifies which are correct loop closures. Each threshold value, as indicated by the red lines, accurately removes a number of candidate loop closures. While not all correct loop closures are selected, very few, if any, incorrect loop closures are kept. These threshold values are determined by experimentation for each of the textures.

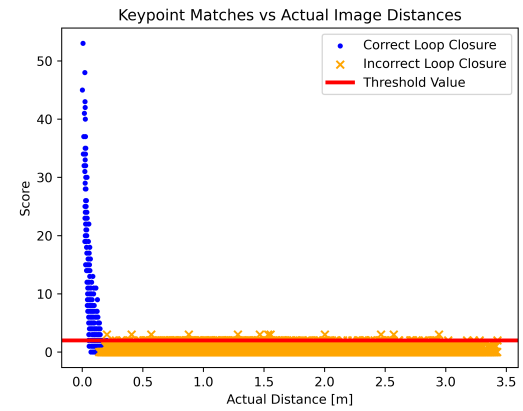
Combining all three thresholds produces highly reliable loop closure selection. Fig. 7 shows an example result, with red lines joining each identified loop closure. Note that each joins two nearby images without false positives. Additionally, Fig. 8 shows a plot of the estimated distance of each selected loop closure versus the actual distance according to the ground truth for selected paths. In almost all cases, the estimated distance is very close to the actual, although some textures have outliers, including one which has been removed for clarity. This means that the proposed system is effective at both identifying loop closures and accurately measuring them for use in drift correction. Fig. 5 illustrates an example of this by showing both our system and our system without any loop closure.

Another result of note is the performance when the texture surface differs from when a map was created, such as after rain. Micro GPS compares captured images to an *a priori* map for localization. Between capturing the map and capturing the images, rain changed the appearance of the ground on select textures, resulting in poor localization, as shown in Table I. Since our SLAM system does not use an *a priori* map, it is only comparing images of textures that are already wet, not comparing between wet and dry. This leads to more consistent performance. However, this may not hold true for very long duration SLAM sessions. Future work will explore those situations.

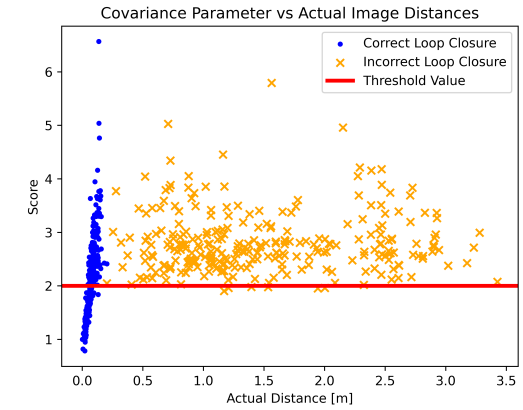
In addition to accuracy, we also timed how long the system ran in each sequence. Fig. 9 reports the results. It shows the number of images in a sequence and average processing



(a) Bag of Words Scores



(b) Keypoint Matches



(c) Covariance Scores

Fig. 6: The three threshold scores for pairs of images vs. the real world distance between those images. Only pairs of images with low actual distance can overlap and form valid loop closures, as shown. For (a) and (b) higher scores indicate better matches. For (c) a lower score indicates a better match. Candidate loop closures that do not meet the threshold are removed from consideration.

time per image within the sequence, which approximates how fast images can be captured. These values are calculated by running a Docker container on a Windows 10 computer with an Intel i7 processor and 64 GB of RAM. The images are 1600 x 1200 pixels. Micro GPS values are also included for

Texture	Path	Normalized Translational MAE (cm/m)		Rotational MAE (deg)	
		Our SLAM	Micro GPS	Our SLAM	Micro GPS
Footpath Test Square	test_path1	5.07	**0.02**	12.11	**0.41**
Footpath Test Square	test_path2	1.96	**0.01**	5.49	**0.35**
Footpath Test Square	test_path_wet1	**1.10**	27.07	**2.56**	114.68
Footpath Test Square	test_path_wet2	**2.15**	24.42	**5.78**	107.29
Parking Place	regular_test_pp2_201010	**1.62**	46.80	**7.22**	71.83
Parking Place	regular_test_pp2_210225_clean_slightly_wet	**6.60**	24.73	**19.14**	30.21
Parking Place	regular_test_pp4_210225_slightly_wet	**2.17**	4.58	13.68	**11.73**

TABLE I: A comparison of translational mean absolute error (MAE), normalized by overall path length, and rotational MAE between our SLAM system and Micro GPS [18]. Sequences shown are highlighted because of changing appearance due to environmental effects.

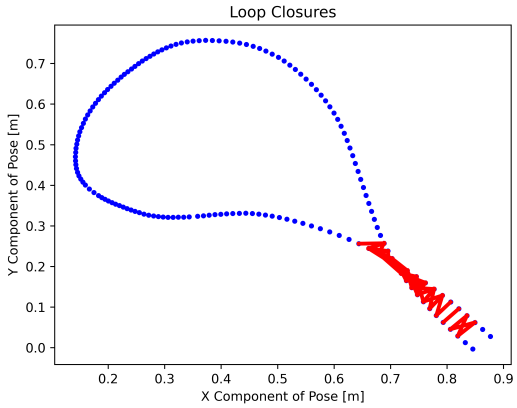


Fig. 7: The loop closures, shown in red, identified during a representative trajectory, with robot poses shown in blue.

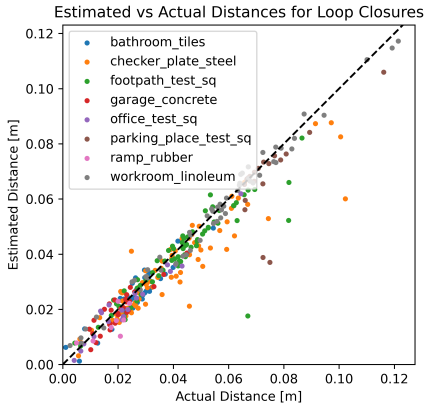


Fig. 8: A plot of various estimated transforms between valid loop closure images pairs. One path from each texture is used for clarity. A y=x line indicating the ideal result is shown for reference.

reference, but do not include the time required to collect and pre-process map data. This map-building time represents a significant duration beyond just runtime that our method does not require.

Notably, our system shows faster performance for the sequences tested due to the rapid candidate loop closure threshold evaluation. However, the data suggests longer sequences may prove unwieldy due to increasing times. Future work will look at map pruning to reduce time.

With both reliable accuracy and fast operating time, the proposed SLAM system provides an effective means of navigation over varied ground textures. Future work aims to realize a system that is more generalizable across a broad

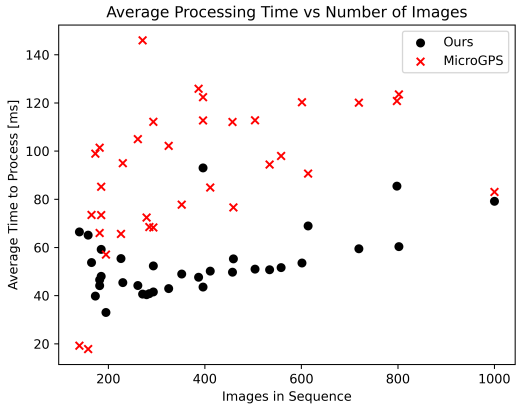


Fig. 9: A plot of the processing speed of our full SLAM system and the Micro GPS localization system for various sequences.

range of textures, is robust to environmental changes during multi-session SLAM, and is fast for long duration sessions.

VI. Conclusions

We have presented an innovative ground texture SLAM system that operates using only a calibrated, downward facing monocular camera. This system is the first to offer full online SLAM capabilities in the ground texture domain without the need for an existing map. When receiving a new image, it detects keypoints in the image and projects them onto the ground plane using known information about the camera position. It then uses robust M-estimator methods to estimate the ground plane 2D transforms experienced by the robot between pairs of images. Loop closures use three threshold values to identify revisited areas to improve overall accuracy. We have presented experimental results showing reliable performance on various ground textures and identified several that require further study to support. We have also shown accurate loop closure identification. When operating on acceptable textures, this system provides a means to rapidly set up robot navigation without the need to provide an *a priori* map.

Acknowledgements

This work was supported in part by NSF Grant IIS-1652064, and by the U.S. Naval Air Warfare Center - Aircraft Division's Naval Innovative Science and Engineering program. We would like to thank J. Fabian Schmid for giving us access to the HD Ground dataset.

REFERENCES

[1] J. F. Schmid, S. F. Simon, and R. Mester, "Features for ground texture based localization - a survey," in *30th Brit. Mach. Vision Conf.*, Cardiff, UK, Sep 2019.

[2] X. Chen, A. S. Vempati, and P. Beardsley, "StreetMap - mapping and localization on ground planes using a downward facing camera," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Madrid, Spain, Oct 2018, pp. 1672–1679.

[3] K. Kozak and M. Alban, "Ranger: A ground-facing camera-based localization system for ground vehicles," in *Proc. IEEE/ION Position, Location and Navigation Symp.*, Savannah, GA, USA, Apr 2016, pp. 170–178.

[4] T. Shan, B. Englot, C. Ratti, and D. Rus, "LVI-SAM: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping," in *Proc. IEEE Int. Conf. Robot. and Automat.*, Xi'an, China, May 2021, pp. 5692–5698.

[5] D. Cheng, H. Shi, A. Xu, M. Schwerin, M. Crivella, L. Li, and H. Choset, "Visual-laser-inertial SLAM using a compact 3D scanner for confined space," in *Proc. IEEE Int. Conf. Robot. and Automat.*, Xi'an, China, May 2021, pp. 2450–2456.

[6] J. Zubizarreta, I. Aguinaga, and J. Montiel, "Direct sparse mapping," *IEEE Trans. on Robot.*, vol. 36, no. 4, pp. 1363–1370, May 2020.

[7] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: an efficient alternative to SIFT or SURF," in *Proc. IEEE Int. Conf. Comput. Vision*, Barcelona, Spain, Nov 2011, pp. 2564–2571.

[8] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. of Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov 2004.

[9] Y. Zhang and J. Leonard, "A front-end for dense monocular SLAM using a learned outlier mask prior," in *Proc. IEEE Int. Conf. Robot. and Automat.*, Xi'an, China, May 2021, pp. 11 732–11 738.

[10] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct 2015.

[11] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct 2017.

[12] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, Dec 2021.

[13] M. Quan, S. Piao, Y. He, X. Liu, and M. Qadir, "Monocular visual SLAM with points and lines for ground robots in particular scenes: parameterization for lines on ground," *J. of Intell. and Robotic Syst.: Theory and Appl.*, vol. 101, no. 4, Mar 2021.

[14] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, "PL-SLAM: real-time monocular visual SLAM with points and lines," in *Proc. IEEE Int. Conf. Robot. and Automat.*, Singapore, May 2017, pp. 4503–4508.

[15] J. Civera, A. Davison, and J. Montiel, "Inverse depth parametrization for monocular SLAM," *IEEE Trans. Robot.*, vol. 24, no. 5, pp. 932–945, Oct 2008.

[16] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *Proc. IEEE Int. Conf. 3D Vision*, Stanford, California, USA, Oct 2016, pp. 239–248.

[17] U.-H. Kim, G.-M. Lee, and J.-H. Kim, "Revisiting self-supervised monocular depth estimation," in *Robot Intell. Technol. and Appl. 6 - Results 9th Int. Conf. Robot Intell. Technol. and Appl.*, Mar 2021, pp. 336–350.

[18] L. Zhang, A. Finkelstein, and S. Rusinkiewicz, "High-precision localization using ground texture," in *Proc. IEEE Int. Conf. Robot. Automat.*, Montreal, QC, Canada, May 2019, pp. 6381–6387.

[19] J. F. Schmid, S. F. Simon, and R. Mester, "Ground texture based localization: do we need to detect keypoints?" in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Las Vegas, NV, USA, Oct 2020, pp. 4575–4580.

[20] L. Zhang and S. Rusinkiewicz, "Learning to detect features in texture images," in *Proc. IEEE/CVF Conf. Comput. Vision and Pattern Recognit.*, Salt Lake City, UT, USA, Jun 2018, pp. 6325–6333.

[21] M. Muja and D. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *Proc. 4th Int. Conf. Comput. Vision Theory and Appl.*, Lisboa, Portugal, Feb 2009, pp. 331–340.

[22] G. Bradski, "The OpenCV library," *Dr. Dobb's Journal of Software Tools*, 2000.

[23] F. Dellaert, "Factor graphs and GTSAM: a hands-on introduction," Georgia Institute of Technology, Atlanta, GA, USA, Tech. Rep. GT-RIMCP&R-2012-002, Sep 2012.

[24] D. Galvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Trans. on Robot.*, vol. 28, no. 5, pp. 1188–1197, Oct 2012.

[25] J. F. Schmid, S. F. Simon, R. Radhakrishnan, S. Frintrop, and R. Mester, "HD ground - a database for ground texture based localization," in *Proc. IEEE Int. Conf. Robot. and Automat.*, Philadelphia, PA, USA, May 2022, pp. 7628–7634.