2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Accurate and Robust Teach and Repeat Navigation by Visual Place Recognition: A CNN Approach

Luis G. Camara*, Tomáš Pivoňka*, Martin Jílek, Carl Gäbert, Karel Košnar and Libor Přeučil

Abstract— **We propose a novel teach-and-repeat navigation system, SSM-Nav, which is based on the output of the recently introduced SSM visual place recognition methodology. During the teach phase, a teleoperated wheeled robot stores in a database features of images taken along an arbitrary route. During the repeat phase or navigation, a CNN-based comparison of each captured image is performed against the database. With the help of a particle filter, the image associated with the most likely location is selected at each time and its horizontal offset with respect to the current scene used to correct the steering of the robot and to navigate. Indoor tests in our lab show a maximum error of less than 10 cm and excellent robustness to perturbations such as drastic changes in illumination, lateral displacements, different starting positions, or even kidnapping. Preliminary outdoor tests on a 0.22 km route show promising results, with an estimated maximum error of less than 25 cm.**

MULTIMEDIA MATERIAL

A video accompanying this paper can be found at http://imr.ciirc.cvut.cz/Downloads/Videos

I. INTRODUCTION

Teach-and-repeat (T&R) navigation [1]–[3] is one of the basic tasks in mobile robotics. As the name suggests, it consists of two parts: (1) a teach phase, usually performed manually by navigating a robot along a desired trajectory while storing information about it and (2) a repeat phase, during which the robot must navigate autonomously and follow the learned path as accurately as possible.

Such navigation systems are built on a variety of sensors, e.g., cameras [1]–[3], LIDARs [4], wheel encoders [5] or GPS [6]. Unfortunately, LIDARs are not always available due to their high cost whereas many working environments such as indoor areas are GPS-denied. A problem with wheel odometry is that it suffers from integration errors caused by drifts and/or motion model inaccuracies, which makes recovering the correct global position a challenging task. This can nonetheless be mitigated to some extend by fusing with other sensors to improve reliability and precision.

On the other hand, the use of cameras has a number of advantages, such as their low cost and high availability as well as the fact that they can be used both indoors and outdoors. Even the most simple robots are very often equipped with a camera.

*Authors have contributed equally.
Authors are with the Czech Institute of Informatics, Robotics and Cybernetics (CIIRC), Czech Technical University in Prague, 160 00 Prague, Czech Republic (e-mail: luis.gomez.camara@cvut.cz; pivontom@fel.cvut.cz; martin.jilek.2@cvut.cz; carl.gaebert@informatik.tu-chemnitz.de; karel.kosnar@cvut.cz; libor.preucil@cvut.cz)

The T&R approach is suitable for plenty of applications, especially those with a static, repetitive task. It can be used, for instance, in autonomous regular inspections or patrolling [7], a task that is determined either by a path or by a set of points that must be checked. If the order of the points or the path itself is static, it is sufficient to repeat a once learned path. Other possible applications, to name a few, are guiding transportation robots in industry [4], navigation of underground mining vehicles [8], or a robotic tourist guide [1].

In this paper, we propose a new T&R navigation system for a wheeled robot. It utilizes a monocular camera together with wheel odometry. A main part of the system consists of a visual place recognition methodology [9]–[11] that is used both for robot localization along the taught trajectory and visual servoing. The place recognition task searches a database containing image descriptors stored during the teach phase and tries to find the image that is most similar to the current scene during repeating. Ideally, the best match represents the position of the robot in the taught path. Since this assumption is not always true, the robustness of recognition is statistically improved by a particle filter, which tracks the robot's most likely position. A steering correction is then computed purely from the horizontal offset between the current camera image and the database image selected by the filter.

The remaining of this paper is structured as follows. The current state of the art in vision-based T&R navigation is presented in Section II. In Section III, we characterize the proposed system and its workflow, whereas Section IV describes the experimental setup and presents tests results in both indoor and outdoor conditions. Finally, Section V is devoted to conclusions and future lines of work.

II. RELATED WORK

Most visual T&R navigation methods can be classified into either position-based (quantitative) or appearance-based (qualitative) approaches [12], [13]. The former directly attempt to find the location of a robot in a map, whereas the latter do not create a map of the environment but merely store sensory snapshots in a database during the teach phase; the robot is then controlled based on a comparison of current sensor information with the best matching database snapshot.

The ability of position-based systems to follow multi-kilometer trajectories was demonstrated in [2]. The reported system performed fully simultaneous localization and mapping (SLAM) using a stereo camera, creating a global map of the environment during teaching. Localization proceeded

by matching the relative 3D position of SURF features [14] with those stored in the map. The robustness of the system to changes in the environment was improved by teaching new features during repeated traversals [15] and by selecting, using the Bag-of-Words model (BoW) [16], only those features from the database that were similar to the current illumination and weather conditions [17].

It is interesting to note that in many other applications, the system is not restricted to visual information only. A recent example is the T&R aerial robot navigation system introduced in [18], which combines stereo vision with inertial measurements for accurate pose estimation.

While the approaches mentioned above are position-based, the method proposed in this paper fits into the appearance-based category. A common and essential task during navigation on this type of system is to control the heading of the robot. In the particular case of terrestrial robots, steering is typically realized according to the horizontal relative displacement of matched features between stored and current images. The first such system was presented in [19], and its improved version using odometry in [1]. During the teach phase, it stores descriptors of detected features as well as their horizontal positions within the images. However, only specific locations that correspond with the first and last images of a number of segments along the path are considered. For each particular segment, odometric information is also saved. Upon initialization of the repeating phase, the robot assumes its position at the first stored segment. At the beginning of any other segment, features' positions that match the current ones are tracked until the end and the horizontal differences used to compute the heading correction. The main advantage of this approach is that it uses an uncalibrated monocular camera.

Although more tied to odometry, a similar approach was presented in [3], where a robot repeated the learned trajectory from compass and odometric information while correcting the heading using visual data. The authors proved that, at least for closed polygonal trajectories, the system did not diverge. The proof was generalized to all types of paths in [5], where an enhanced T&R method known as Bear-Nav was also presented. In the latter, heading during navigation is corrected using the horizontal offset of matched features, but recorded velocities are also repeated. As shown in Section IV, this method will be used as the baseline against which to compare our approach.

In recent years, the popularity and excellent performance of neural networks in computer vision applications have also transcended into robotics and T&R navigation is not an exception. In [20], the authors presented a deep visual T&R framework for navigating in routes that contained simple intersections. They also introduced an unsupervised labeling scheme using monocular visual odometry for image sequences. After training on images from the teach path, a neural network was able to return, given an input image, discrete steering commands (forward, left and right). In [21], objects detected and classified by a Convolutional Neural Network (CNN) were directly used as features, which were

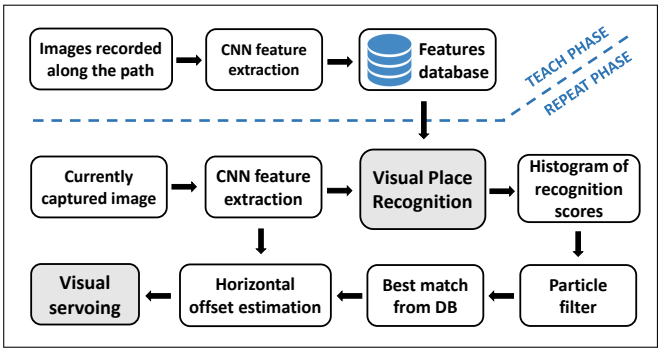

Fig. 1: Flowchart of the visual navigation system.

characterized by the region where they appeared within the image and by the type of object they described (e.g., chair, monitor). A steering correction was then computed directly from the horizontal displacement of matched features, similarly to [1].

The system presented here is also based on semantic information and horizontal offsets between features. However, rather than being explicitly expressed as objects, we employ high-level features from the VGG16 CNN architecture, which as shown in [9]–[11] can lead to very robust image representations.

III. Teach-and-Repeat Methodology

The flowchart in Fig. 1 summarizes the vision part of the proposed navigation system. During the teach phase, a mobile robot (see Sec. IV-C) is guided (taught) by teleoperation along an arbitrary path, with pictures of the scene being taken regularly. Images are then passed through a CNN, resulting in a set of features that are stored in a database (see [9]–[11] for details). During the repeat phase, captured images are processed in the same fashion as during teaching and the generated features compared against the database to find matches. This is carried out in real-time by the visual place recognition system (Section III-A). Rather than choosing the best matching image only, a number of candidates are considered and their strength as potential matches expressed in a histogram of recognition scores. The histogram is then fed into a particle filter (Section III-B), which in combination with a motion model estimates the best matching place at each time. Next, the image in the database corresponding to that place is geometrically compared with the image of the current scene. The comparison tries to find a consistent horizontal offset in the spatial locations of matching (closest) features in both images. The obtained offset serves as visual servoing to send steering signals to the robot's actuators.

Note that our system does not create a map of its environment nor repeats recorded velocities during navigation. It only stores representations of images captured along a taught path and applies visual servoing to correct the heading.

A. Visual place recognition

As seen above, the core of the navigation system presented in this work is based on visual place recognition. We have

Fig. 2: Left: Husky A200 robot used throughout this work. Right: passing through a narrow door during navigation.

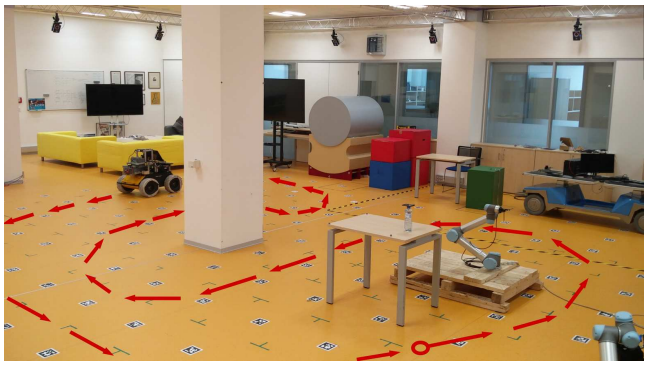

Fig. 3: Teach path for indoor tests in our lab. Red circle denotes the starting location.

implemented the methodology known as SSM-VPR (Semantic and Spatial Matching Visual Place Recognition) [9]–[11], which has recently shown state-of-the-art performance in this task. It consists of a two-stage pipeline that encodes images by creating descriptors from the activations of a pre-trained CNN architecture (VGG16). During stage I, the system retrieves from a database of images a list of N candidates which are perceptually closer to a given query image. In stage II, the candidates are compared one by one with the query. Through an intense geometrical consistency check (spatial matching), a histogram of recognition scores is constructed for the candidates. The candidate with the highest score is commonly selected as the recognized place although, as shown below, the entire histogram can be used to make a more robust decision.

B. Particle filter

During place recognition, there are usually several images among the list of candidates that represent places in the vicinity of the robot's location. Since these images may be perceptually similar to each other, their score in the candidate's histogram will also be comparable. It may be the case, for instance, that a candidate whose location is a few meters past the current place looks nonetheless more similar to that place than a spatially closer candidate. Factors such as occlusions or differences in viewpoint may be behind this. An additional source of ambiguity comes in the form of perceptual aliasing caused by the existence of repetitive structures. In this case, even candidates that are far away from the current place may show up as good potential matches. Strictly relying on the best match could therefore introduce errors of up to the total length of the path.

In order to deal with the aforementioned issues, we have implemented a particle filter [22] that steadily estimates the state x_t of the robot and, at the same time, deals with potentially spurious candidates. Particles are defined as a set of sample states $\{x_t^{[i]}\}$, where i denotes the index of each particle and runs from 1 to M, the size of the filter. Each particle $x_t^{[i]}$ represents a possible position along the path, projected to a single dimension by referring it to the distance traveled from the starting location. At every filtering step, the motion model given by Eq. (1) updates those positions by adding to each particle the distance d traveled by the robot since the last measurement, obtained from wheel odometry.

$$x_t^{[i]} = x_{t-1}^{[i]} + d \; ; \;\; i \in \{1, 2, \dots M\} \tag{1}$$

In addition to motion, a sensor model is used to update the individual weight of each particle, $w_t^{[i]}$, which reflects how much the particle's location relates to the current place. During the initialization of the filter at the starting position, locations are randomly generated from a uniform distribution and the weights set to M^{-1}. The model is given by the following expression:

$$w_t^{[i]} = w_{t-1}^{[i]}(1 + \hat{s}^{[i]}) \; ; \;\; i \in \{1, 2, \dots M\}, \tag{2}$$

where the term $\hat{s}^{[i]}$ represents the recognition score assigned to particle i and normalized over all candidates' scores. Each particle is assigned the score of its closest location's image, which is zero for locations other than those of the N candidates. During this process, stored odometry is indirectly used through the particle filter, as the score assignment requires knowledge about the location of each database image. After applying Eq. (2), all weights are normalized. Thus, particles that are near the location of candidates become more likely to be the actual position of the robot, whereas other particles' weights tend to decrease. In a final step, the 10% of particles with the lowest weights as well as those that exceed the total distance of the route are removed. They are replaced by generating uniformly distributed particles over the whole path. In this fashion, the filter is able to converge to one location while keeping the ability to change upon new measurements. The actual position of the robot along the path is estimated by averaging over the positions of the most heavily weighted particles, whose optimal number can be determined experimentally.

C. Visual servoing and navigation speed

The key for VPR-based navigation with the proposed system is its ability to estimate a consistent horizontal offset between the on-board camera image and the matching database image. This offset is used for visual servoing (VS) the robot by the right measure and in the direction that would horizontally align the two images. To calculate it, a histogram

TABLE I: Average maximum error over 3 laps for indoors tests.

System	Illumination	Candidates	Max. error (m)
SSM-Nav (VGG16)	Bright	5	**0.066**
"	Dark	10	0.116
"	Very dark	10	0.148
SSM-Nav (NetVLAD)	Bright	5	0.096
"	Dark	10	0.078
"	Very dark	Failed	Failed
Bear-Nav [5]	Bright	–	**0.107**
"	Dark	–	0.147
"	Very dark	–	0.291

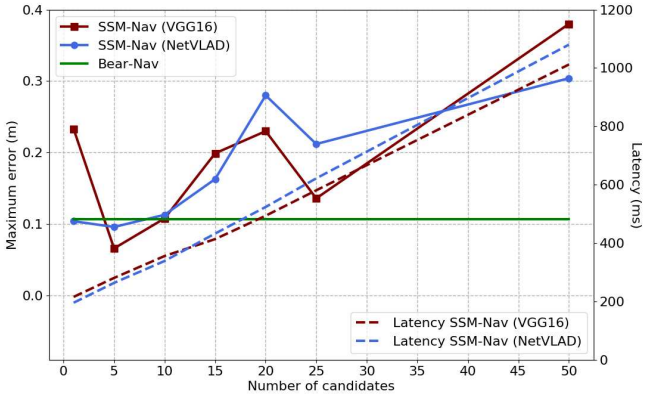

Fig. 4: Navigation error as a function of the number of candidates under normal (bright) illumination conditions and for the SSM-Nav (VGG16) and SSM-Nav (NetVLAD) systems. Latency is also displayed (scale on right vertical axis). The horizontal line is the error associated with the Bear-Nav [5] navigation system.

of horizontal differences is accumulated by looking at the relative positions of matching descriptors in both images, with the final highest bin in the histogram representing the offset. The reader is referred to [11] for a full description of the spatial matching process.

Regarding the robot's linear speed, a maximum value of 0.35 m/s was set during our indoor experiments on straight trajectories. Otherwise, the speed was proportionally reduced by the amount of turning applied in the previous step. In this fashion, sharp turns were safely traversed at a lower speed, which helped to reduce navigation error.

IV. Experiments and Results

A. Indoor tests

Our lab is equipped with a Vicon motion capture system consisting of 20 infrared cameras and capable of measuring a robot's position with an accuracy of less than a millimeter. We have used it to measure the error between teach and repeat paths in a number of indoor tests. The robot was initially guided with a joystick along a closed path of approximately 35 m, as illustrated by the red arrows in Fig. 3. The trajectory along the path was stored for comparison with a range of repeat experiments, which are described below.

1) System's latency and number of candidates: The latency or time required by the process shown in Fig. 1 during the repeat phase is a fundamental parameter to consider during navigation, as frequent feedback is needed by the robot to make appropriate steering decisions. Equally important is N, the number of candidates from the database that are compared with the current image in stage II during recognition. A list of candidates that is too small may deteriorate localization, since the best geometrical match may not be in that list. On the other hand, utilizing more candidates may increase recognition accuracy but at the cost of slowing down the entire process.

Our first experiments focused precisely on assessing the interplay between these two parameters. Tests were performed under optimal (bright) illumination conditions. Results of the analysis are depicted in Fig. 4, which shows the maximum error (the separation from the teach path) averaged over 3 laps and when taking into account from 1 to 50 candidates. Latencies in milliseconds are also shown on the vertical scale on the right. As in [10], we considered two versions of the recognition system, one using NetVLAD [23] and the other using VGG16 as the architecture in stage I during recognition. As can be seen, the general behavior is the improvement in navigation accuracy as the latency is decreased, although using less than $N=5$ candidates start to affect performance negatively. Thus, it seems that using this figure provides the best trade-off, with a latency of around 250 ms and a maximum error below 10 cm in both cases. With such low error, we were able to successfully navigate the robot through narrow doors and corridors (see Fig. 2).

For comparison, Fig. 4 also shows maximum error of the Bear-Nav [5] T&R system (green line), which as mentioned earlier in this paper uses a combination of visual servoing, stored velocities and odometry for navigation. We note that the latency of the Bear-Nav system allowed for real time image processing (e.g. $\geq$25 fps). Even though SSM-Nav does not rely on velocities, the results are comparable or even better than Bear-Nav on these tests. The actual values are shown in Table I under *bright* illumination conditions.

2) Illumination conditions: Along with differences in viewpoint, illumination variability is one of the most challenging condition changes for a VPR system. As can be appreciated in Table I, we tested navigation under three different conditions: *bright*, *dark* and *very dark*. Example images of each of them are shown in Fig. 5, with perspectives from one corner of the lab and from the robot's on-board camera. For the *bright* case, all lights in the lab were switched on, whereas for the *dark* one they were all turned off, and only illumination from an outside corridor was made available. Under *very dark* conditions, the lab was set as dark as possible and a led torch attached to the front of the robot as otherwise, navigation was not feasible.

Table I summarizes the best results from the experiments, with the actual 3-lap trajectories depicted in Fig. 6. The *bright* case has already been considered in Fig. 4, where the SSM-Nav (VGG16) showed the best performance with less than 7 cm of maximum displacement from the teach path. For the *dark* set up, the lowest error was achieved only when the number of candidates was increased to $N = 10$. On this oc-

Fig. 5: Examples of the different illumination conditions considered during indoor tests: *bright*, *dark* and *very dark*. Left column: lab view. Right column: on-board camera view.

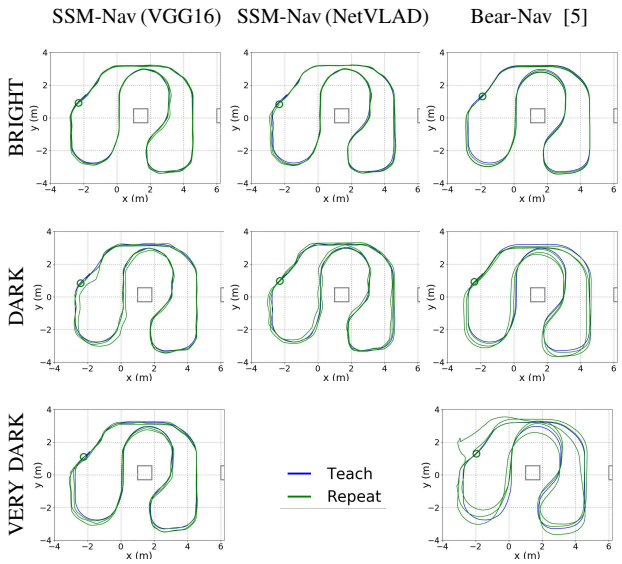

Fig. 6: Repeat traverses of the SSM-Nav and Bear-Nav [5] systems for the indoor tests shown in Table I.

casion, the NetVLAD version performed better than VGG16, with Bear-Nav being the worst of the three. Under *very dark* conditions, tests on the NetVLAD version failed, with the robot getting lost even when increasingly larger number of candidates were tested. The VGG16 version, however, showed excellent performance considering the difficult conditions, with around 15 cm of maximum error. These results are compatible with those in [10], [11], where it was shown that descriptors extracted from pre-trained networks could be more robust than those taken from customized networks, at least in cases where training sets are not representative of test samples. The NetVLAD model utilized here was trained on a day-only, outdoors urban dataset, hence its bad performance in very dark environments. On the other hand, Bear-Nav was not able to complete the *very dark* test fully autonomously and needed assistance in correcting the trajectory on one occasion. Even so, it often departed significantly from the teach path, as can be appreciated in Fig. 6. The final average maximum error was close to 30 cm, almost two times that of our system.

3) Lateral displacements: Using the configuration with less error from Table I (SSM-Nav (VGG16); *bright*; 5 candidates), the robot was sent to repeat the path and its robustness to sudden, lateral displacements of up to 0.8 meters tested. First image in Fig. 7 illustrates how it recovered from such disruptions. As can be seen, convergence towards the teach path occurred in all cases and within a distance of 2 to 4 meters, showing the good stability of the system upon this sort of events.

4) Starting location: The ability of the robot to start navigation at arbitrary locations along the route was also evaluated. This is a more challenging test than the lateral displacement discussed above, as the particle filter is not yet initialized and localization must be started afresh. A number of locations were chosen, such that the robot was looking in the direction and sense of the route, although not necessarily parallel to it. We did not consider locations that were exactly on the path since they always led to successful localization. Results are depicted in Fig. 7 for 11 initial locations. One can see how the robot manages to join the path in most cases, sometimes straight away and sometimes not so rapidly. The only unsuccessful test (in red) was due to the robot heading towards the central pillar of the room (grey square), most likely when trying to join the path on the opposite side. We are planning to implement an obstacle avoidance system based on stereo vision that is expected to cope with these situations.

5) Kidnapping: In order to make tests even more challenging, we experimented with difficult initial locations and headings. In our implementation, this is similar to the *robot kidnapping problem*, as the latter can potentially place the robot at such difficult conditions and force a restart in the localization process. We begun by locating the robot near the four corners of the central pillar and heading towards the exterior of the room. Resulting trajectories are illustrated by the 4 images on the left column of Fig. 8. Under these conditions, the camera points, roughly, either perpendicularly to the closest path (first, second and fourth image) or in a similar direction but opposite sense (third image). As can be appreciated, the robot successfully joined the path in all cases. Four more tests were carried out by locating the robot at the four corners of the lab's working area, also heading towards the exterior of the room and therefore not being able to see the path. This is depicted in the central column of Fig. 8. The four tests were again successful, which implied that the robot had to turn around completely to follow the path.

The right column in Fig. 8 shows the results of 4 tests where the locations and headings were chosen randomly. On this occasion, the second and fourth failed due to the robot navigating outside of the working area.

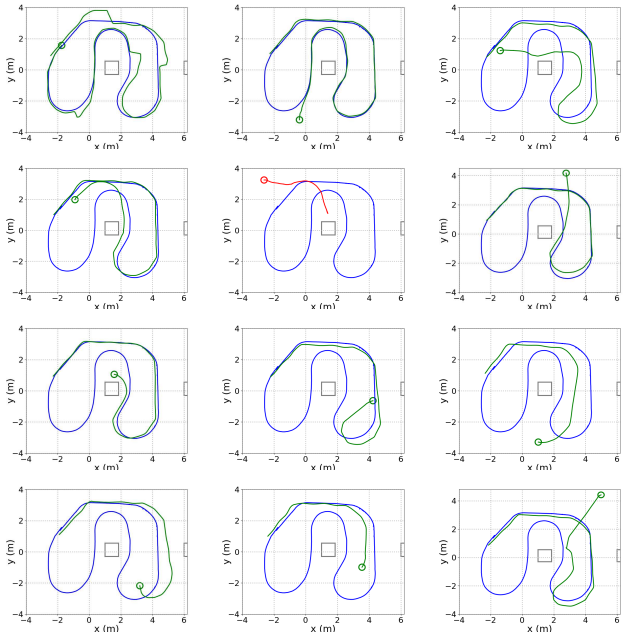

Fig. 7: Top left image: lateral displacement experiments. Rest of images: experiments on different starting locations, denoted by an open circle. The red path was unsuccessful due to the robot heading towards the central pillar (grey square).

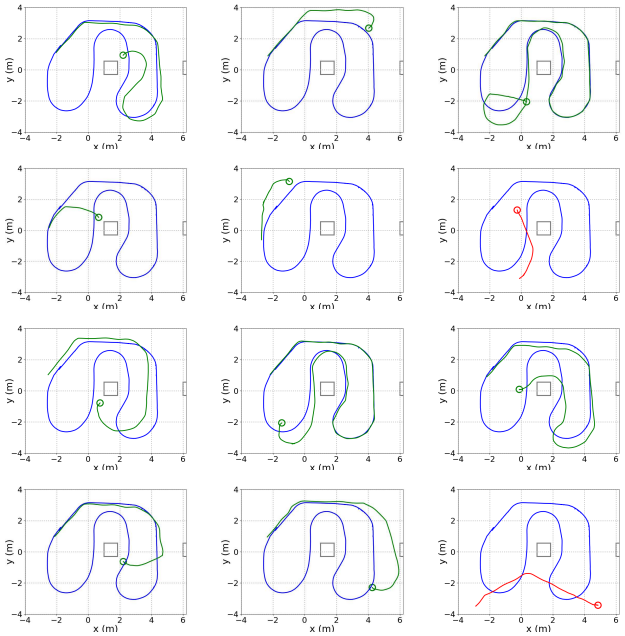

Fig. 8: Tests under difficult initial conditions (analogous to kidnapping). Left column: robot located at the four corners of the central pillar (grey square) and looking outwards. Central column: robot located at the four corners of the lab and looking outwards. Right column: robot located at random positions and headings. Red curves represent tests where localization/recovery failed.

Currently, a rather simplistic scheme is implemented to start navigation. Based on the output of the particle filter, whenever a sufficiently prominent and narrow peak is observed around a location in the path, the robot is considered localized. Peaks are characterized by comparing location and weight statistics of the peak's particles with those of the rest of particles. Once localization is detected, navigation proceeds normally. Otherwise, the robot's linear speed is kept very low, so that it has time to build up knowledge about the current location. If localization fails after some time, the robot is steered to the left (which may be especially favorable for this particular teach path) by a random amount and localization attempted again. We expect that slightly more sophisticated and general approaches will improve robustness to difficult initial conditions such as these.

B. Outdoor tests

Preliminary tests outside of our lab's building were carried out in the surrounding campus. The linear speed of the robot was set to 0.5 m/s on straight lines and automatically decreased during turns, as described earlier in this paper. GPS data for the SSM-Nav (VGG16) repeat traverse is shown in Fig. 9. It was 220 m long and consisted of a combination of paved and unpaved paths, with some sharp corners and uneven terrain. Because insufficient precision of the utilized GPS, we did not use it to test the repeat phase accuracy. Instead, we manually measured the error between the teach and repeat trajectories at 8 specific locations along the route (see small red crosses in the map). The precision of the measurement was roughly estimated to be under 7cm. Both systems were able to successfully complete the circuit, with average errors of 15 cm for SSM-Nav (VGG16) and 12 cm for SSM-Nav (NetVLAD). Maximum errors were 38 cm and 24 cm, respectively. We are planning further tests where a more accurate tracking system such as a differential GPS will be used but so far, current results look very promising.

C. Hardware and software

All experiments were carried out on a Clearpath Husky A200 UGV (Fig. 2). The robot was equipped with an Intel RealSense D435 camera with a resolution of 1920x1080 px and a Zotac Zbox Magnus mini EK51070 computer, equipped with an Nvidia GeForce GTX 1070 graphics card. The operating system was Ubuntu 16.04 and the whole control software was written using ROS Kinetic. Low-level control routines were written in C++, whereas the navigation algorithm utilized Python 2.7 and the Keras neural network library.

D. Memory usage

Memory utilized by SSM-Nav during the lab tests was approximately 2 GB of RAM for both VGG16 and NetVLAD versions, whereas Bear-Nav used 250 MB for the same tests. Database descriptors took 360 MB of disk space for SSM-Nav and 880 MB for Bear-Nav.

V. Conclusions

In this paper, we have shown that accurate and robust teach-and-repeat navigation can be achieved by means of visual place recognition.

Under good illumination conditions, indoor experiments demonstrate that the proposed system is able to repeat an arbitrary path with an average maximum error of less than

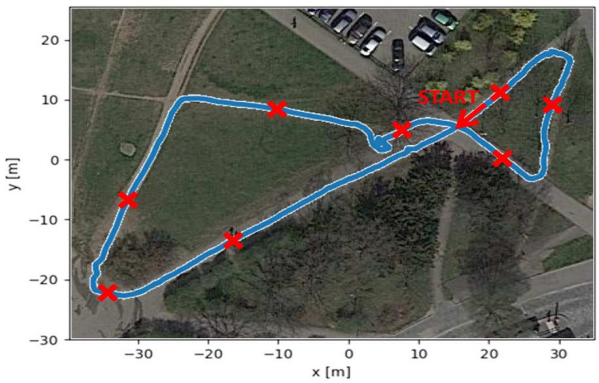

Fig. 9: GPS data of the repeating phase for the SSM-Nav (VGG16) on the outdoor tests. Crosses represent estimated positions of measured locations. Total length of the traverse was 220 m. Map courtesy of Google Maps [24].

7 cm, which is sufficiently small to allow a robot to navigate through doors and corridors inside buildings. Furthermore, the navigation system shows excellent robustness to physical perturbations such as sudden lateral displacements from the path, different starting locations, or the robot kidnapping problem. Challenging tests under poor and very poor illumination conditions confirm the excellent behavior and robustness of the utilized CNN-based image descriptors.

During preliminary outdoor tests, we were able to successfully navigate along traverses of more than 200 m with a maximum error of less than 25 cm.

Work is in progress to refine the system, to make it more robust to the kidnapping problem and to compare it with more general navigation approaches such as ORBSLAM2 [25]. It is also our intention to continue with the outdoor experiments and to test the system against accurate differential GPS measurements as well as poor illumination conditions during the night.

Acknowledgment

This work has been supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 688117, the European Regional Development Fund under the project Robotics for Industry 4.0 (reg. no. CZ.02.1.01/0.0/0.0/15 003/0000470), the Grant Agency of the CTU (No. SGS18/206/OHK3/3T/37) and the Technology Agency of the Czech Republic under the projects No. TN01000024: National Centres of Competence , and No. TH03010369: program Epsilon.

References

[1] Z. Chen and S. T. Birchfield, "Qualitative vision-based path following," *IEEE Transactions on Robotics*, vol. 25, no. 3, pp. 749–754, June 2009.

[2] P. Furgale and T. Barfoot, "Stereo mapping and localization for long-range path following on rough terrain," in *2010 IEEE International Conference on Robotics and Automation*, May 2010, pp. 4410–4416.

[3] T. Krajník, J. Faigl, V. Vonásek, K. Kosnar, M. Kulich, and L. Preucil, "Simple yet stable bearing-only navigation," *J. Field Robotics*, vol. 27, pp. 511–533, 2010.

[4] C. Sprunk, G. D. Tipaldi, A. Cherubini, and W. Burgard, "Lidar-based teach-and-repeat of mobile robot trajectories," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2013, pp. 3144–3149.

[5] T. Krajník, F. Majer, L. Halodová, and T. Vintr, "Navigation without localisation: reliable teach and repeat based on the convergence theorem," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1657–1664.

[6] S. Li and A. Hayashi, "Robot navigation in outdoor environments by using gps information and panoramic views," in *Proceedings. 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications (Cat. No.98CH36190)*, vol. 1, Oct 1998, pp. 570–575 vol.1.

[7] M. Fehr, T. Schneider, and R. Siegwart, "Visual-inertial teach and repeat powered by google tango," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 1–9.

[8] J. Marshall, T. Barfoot, and J. Larsson, "Autonomous underground tramming for center-articulated vehicles," *Journal of Field Robotics*, vol. 25, no. 6-7, pp. 400–421, 2008. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.20242

[9] L. G. Camara and L. Přeučil, "Spatio-semantic convnet-based visual place recognition," in *2019 European Conference on Mobile Robots*. IEEE, 2019, pp. 1–8.

[10] L. G. Camara, C. Gäbert, and L. Přeučil, "Highly robust visual place recognition through spatial matching of cnn features," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.

[11] L. G. Camara and L. Přeučil, "Visual place recognition by spatial matching of high-level cnn features," *ResearchGate*, 2020.

[12] A. Vardy, "Using feature scale change for robot localization along a route," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2010, pp. 4830–4835.

[13] T. Nguyen, G. K. I. Mann, R. G. Gosine, and A. Vardy, "Appearance-based visual-teach-and-repeat navigation technique for micro aerial vehicle," vol. 84, no. 1-4, pp. 217–240, 2016. [Online]. Available: http://link.springer.com/10.1007/s10846-015-0320-1

[14] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*. Springer, 2006, pp. 404–417.

[15] M. Paton, K. MacTavish, M. Warren, and T. D. Barfoot, "Bridging the appearance gap: Multi-experience localization for long-term visual teach and repeat," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 1918–1925.

[16] Sivic and Zisserman, "Video google: a text retrieval approach to object matching in videos," in *Proceedings Ninth IEEE International Conference on Computer Vision*, Oct 2003, pp. 1470–1477 vol.2.

[17] K. MacTavish, M. Paton, and T. D. Barfoot, "Visual triage: A bag-of-words experience selector for long-term visual route following," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 2065–2072.

[18] M. Nitsche, F. Pessacg, and J. Civera, "Visual-inertial teach repeat for aerial robot navigation," in *2019 European Conference on Mobile Robots (ECMR)*, Sep. 2019, pp. 1–6.

[19] Zhichao Chen and S. T. Birchfield, "Qualitative vision-based mobile robot navigation," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, May 2006, pp. 2686–2692.

[20] T. Swedish and R. Raskar, "Deep visual teach and repeat on path networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, June 2018, pp. 1614–161 409.

[21] A. G. Toudeshki, F. Shamshirdar, and R. Vaughan, "Robust uav visual teach and repeat using only sparse semantic object features," in *2018 15th Conference on Computer and Robot Vision (CRV)*, May 2018, pp. 182–189.

[22] S. Thrun, "Particle filters in robotics," in *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 2002, pp. 511–518.

[23] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5297–5307.

[24] "Google maps." [Online]. Available: https://www.google.com/maps

[25] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.