

# Navigation without localisation: reliable teach and repeat based on the convergence theorem

Tomáš Krajník, Filip Majer, Lucie Halodová, Tomáš Vintr

**Abstract**—We present a novel concept for teach-and-repeat visual navigation. The proposed concept is based on a mathematical model, which indicates that in teach-and-repeat navigation scenarios, mobile robots do not need to perform explicit localisation. Rather than that, a mobile robot which repeats a previously taught path can simply “replay” the learned velocities, while using its camera information only to correct its heading relative to the intended path. To support our claim, we establish a position error model of a robot, which traverses a taught path by only correcting its heading. Then, we outline a mathematical proof which shows that this position error does not diverge over time. Based on the insights from the model, we present a simple monocular teach-and-repeat navigation method. The method is computationally efficient, it does not require camera calibration, and it can learn and autonomously traverse arbitrarily-shaped paths. In a series of experiments, we demonstrate that the method can reliably guide mobile robots in realistic indoor and outdoor conditions, and can cope with imperfect odometry, landmark deficiency, illumination variations and naturally-occurring environment changes. Furthermore, we provide the navigation system and the datasets gathered at [www.github.com/gestom/stroll\\_bearnav](http://www.github.com/gestom/stroll_bearnav).

## I. INTRODUCTION

A considerable progress in visual-based systems capable of autonomous navigation of long routes was achieved during the last decade. According to [1], [2], vision-based navigation systems can be divided into map-less, map-based, and map-building based. Map-less navigation systems such as [3], [4], [5] aim to recognise traversable structures (e.g. roads, pathways, field rows, etc.) and use these to directly calculate motion commands. Map-based navigation systems rely on environment models that are known apriori [6]. Map-building-based systems rely on maps for localisation and navigation as well, but they can build these maps themselves. Some of these vision-based methods can build maps and localise the robot at the same time – these are referred to as visual SLAM (Simultaneous Localisation and Mapping).

One of the most known visual SLAM systems, Monoslam [7], processes an image stream from an unconstrained-motion monocular camera in real-time, obtaining the trajectory of the camera and a 3D map of salient visual features [7]. Another method, the ORB-SLAM [8], [9], allows exploiting stereo and depth information to build both sparse and dense maps of the environment while estimating the camera motion in 6D. Unlike the aforementioned systems, LSD-SLAM [10] and DSO [11] do not rely on image feature extraction but create dense, large-scale maps

by directly processing the intensities of the image pixels. A recent, comprehensive review of SLAM systems is presented in [12]. While being an important component of many navigation systems, SLAM by itself does not control the mobile robot motion, and thus it does not navigate robots per se. Rather than that, it provides an environment map and a robot position estimate to the motion planning modules, which then guide the robot towards the desired goal.

Thus, one of the typical use of SLAM methods in practice is ‘teach-and-repeat’, where a robot uses SLAM during a teleoperated drive, creating a map of the environment and use this map later on to repeat the taught path [13], [14], [15]. This technique is analogous to a popular practice in industrial robotics, where an operator teaches a robot to perform some task simply by guiding its arm along the desired path. The systems that use SLAM methods within the teach-and-repeat paradigm were extended by techniques like experience-based localisation [16], feature selection [17] or intrinsic image [18], enabling their long-term deployment in environments that are challenging due to appearance changes [19], [20], [21] or difficult illumination conditions [22].

In the long-term deployments, it’s assumed that the robots start and end their forays at their recharging stations with a known position, and occasional loss of localisation is solved by request for human intervention [23]. Thus, teach-and-repeat methods employed in long-term scenarios do not typically address the kidnapped robot problem.

Some of the teach-and-repeat systems do not rely on SLAM-built 3D maps of the environment, and employ visual servoing principles while respecting robot dynamics [24]. For example [25], [26] create a visual path, which is a set of images along the human-guided route, and then employ visual servoing to guide robots across the locations these images were captured at. Similarly, [27] represents the path as consecutive nodes, each containing a set of salient visual features, and uses local feature tracking to determine the robot steering to guide it to the next node. The authors of [28] extract salient features from the video feed on-the-fly and associate these with different segments of the teleoperated path. When navigating a given segment, their robot moves forward and steers left or right based on the positions of the currently recognised and already mapped features. The segment end is detected by means of comparing the mapped segment’s last image with the current view. The same navigation principle was recently deployed on micro aerial vehicles in [29].

At the time when the original SLAM-based teach-and-repeat framework was published [14], another article [30]

Artificial Intelligence Center, Faculty of Electrical Engineering, Czech Technical University [tomas.krajnik@fel.cvut.cz](mailto:tomas.krajnik@fel.cvut.cz)

The work has been supported by projects 17-27006Y and CZ.02.1.01/0.0/0.0/16.019/0000765.

mathematically proved that (while being useful) explicit localisation is not necessary for teach-and-repeat scenarios. The results of [30] indicate that to repeat a taught path, camera input needs to be used only to correct the robot heading, leaving the position estimation to odometry. The mathematical proof showed that for polygonal paths the heading corrections suppress odometric errors, preventing the overall position error of the robot to diverge. The proof was supported by several long-term experiments [30], [31], where a robot repeatedly traversed long paths in natural environments over the period of one year. Thus, this SLAM-less method showed good robustness to environment changes even without using experience-based or feature-preselection techniques [31]. However, the mathematical proof in [30] was limited to paths consisting of straight segments. Thus, the system based on [30] could be taught only polygonal paths in a turn-move manner, and even a slight change of the movement direction during the teaching phase required to stop and turn the robot, which made the teaching tedious and deployment of the system rather impractical.

In this paper, we reformulate the problem presented in [30] in a continuous rather than a discrete domain. This allows us to simplify and extend the mathematical proof of [30] to any continuous trajectory, not only polygonal paths as in [30]. A navigation system based on this extended formulation can thus be taught smooth, curved paths, which makes its teaching faster and navigation more efficient.

The main contribution of this paper is the aforementioned mathematical proof which indicates that in teach-and-repeat scenarios, a robot can use its camera information only to correct its heading and it does not have to build metric maps or perform explicit localisation. Based on this principle, we implement a teach-and-repeat navigation system and use it to verify the aforementioned hypothesis in realistic conditions. In a series of experiments, we compare the behaviour of the system with the proposed mathematical model and demonstrate the system's ability to reliably guide robots along the taught paths in adverse illumination conditions including night. Furthermore, we present the system as an open-source, ROS-based package and accompany the software with the datasets gathered during the experiments performed [32].

## II. NAVIGATION STABILITY

In this section, we analyse how heading correction influences the overall position error of a robot as it travels along a taught path. At first, we establish a model of the robot movement along the desired path and we outline a model of the robot position error. Then, we examine conditions under which the robot position error does not diverge.

### A. Paths with non-zero curvature

Let us assume that a human operator placed a robot at an initial position  $x_p(0), y_p(0)$  and then she drove the robot by controlling its forward  $v$  and angular  $\omega$  velocities. Let the robot record its  $v$  and  $\omega$  velocities together with the features detected in its camera image and let the robot index the features and velocities by the distance travelled. Thus, the

taught path  $\mathcal{P}$  is defined by the initial point  $x_p(0), y_p(0)$ , velocity functions  $v(d)$ , and  $\omega(d)$ , where  $d$  represents the length of the path from  $x_p(0), y_p(0)$  to the current position. Some locations (again indexed by  $d$ ) on the trajectory are also associated with image coordinates and descriptors of the image features detected in the robot camera image. Since the travelled distance  $d$  depends on the robot forward velocity  $v(d)$  by  $d(t) = \int_0^t v(d(\tau))d\tau$ , one can express the taught trajectory as  $x_p(t) = x_p(d(t))$  and  $y_p(t) = y_p(d(t))$ .

Then, assume that a robot is placed at a position  $x_r(0), y_r(0)$  and it is supposed to traverse the path  $\mathcal{P}$ . The robot, having no information about its position or orientation, has to assume that it is at the start of the taught path, and thus its position estimate is  $x_p(0), y_p(0)$ . Therefore, the robot sets its forward and angular velocity to  $v(0)$  and  $\omega(0)$ , respectively. The robot also retrieves the image features it saw at  $d = 0$ , matches them to the features in its current view and adjust its angular velocity in a way, which would decrease the horizontal distances (in the image coordinates) of the matched feature pairs. As the robot moves forwards, it calculates the distance travelled  $d$  and sets its forward velocity to  $v(d)$  and angular velocity to  $\omega(d) - \alpha\kappa$ , where  $\kappa$  is the most frequent horizontal displacement of the matched feature pairs and  $\alpha$  is a user-set constant dependent on the robot dynamics and camera used.

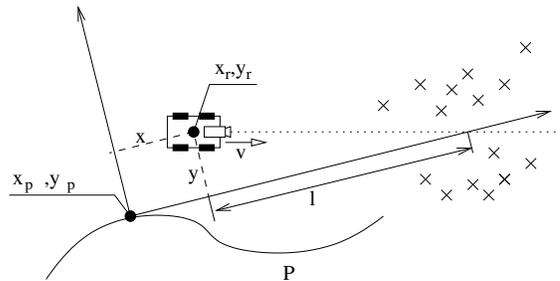


Fig. 1: Robot position error chart. The robot at a position  $x_r, y_r$  uses a local feature map of the taught path  $\mathcal{P}$  at the position  $x_p(t), y_p(t)$ .

Since the robot camera is facing in the direction of the robot movement during the teaching phase, each feature that is in the current map lies in the vicinity of the tangent to the path at some finite distance, see Figure 1. Assuming that the robot is able to turn fast enough to keep  $\kappa$  low (i.e., it is able to turn so that the horizontal distances of the mapped/detected feature pairs are low), the direction of its current movement intersects the aforementioned tangent at a certain distance  $l$ . The distance  $l$  is related to the spatial distribution of the features in the traversed environment – low  $l$  is typical for cluttered environments and high  $l$  occurs mostly in large open areas.

Figure 1 illustrates that the error of the robot position estimate over time  $x(t), y(t)$  can be defined as its position in a local coordinate frame defined by the location and orientation of the local map, which the robot uses to determine its velocities. In other words, the robot position error  $x(t), y(t)$  is

defined by its position  $(x_r(t), y_r(t))$  relatively to  $x_p(t), y_p(t)$ . In order to analyse the evolution of the position error during the robot navigation, we form a differential equation describing  $(\dot{x} = dx/dt, \dot{y} = dy/dt)$ :

$$\begin{aligned} \dot{x} &= +\omega y - v_r + v + s_x \\ \dot{y} &= -\omega x - vyl^{-1} + s_y, \end{aligned} \quad (1)$$

where the terms  $\omega y$ ,  $\omega x$  and  $-v_r$  are caused by the rotation and translation of the local coordinate system as the point  $x_p(t), y_p(t)$  moves along the intended path, the term  $+v$  is the movement of the robot along the path,  $vyl^{-1}$  reflects the influence of the visual-based heading correction, and  $s_x$  and  $s_y$  represent perturbations caused by image processing imperfections, odometric errors and control system inaccuracies. Since the errors  $s_x$  and  $s_y$  directly influence the velocities  $\dot{x}$ ,  $\dot{y}$  of the robot in our model, they encompass not only additive errors, such as occasional wheel slippage or imperfect image feature localisation, but also systematic errors, such as miscalibration of the odometry causing the  $v_r$  and  $v$  to be different, or camera misalignment causing an offset in robot heading corrections. Assuming that  $v_r$  and  $v$  are almost identical, and that their difference is included in the perturbation  $s_x$ , we can rewrite (1) in a matrix form:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} 0 & +\omega \\ -\omega & -vl^{-1} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} s_x \\ s_y \end{pmatrix}, \quad (2)$$

which is a system of linear differential equations.

In general, a linear continuous system  $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{s}$  is stable, i.e., the  $\mathbf{x}$  does not diverge, if the real components of all eigenvalues of the matrix  $\mathbf{A}$  are smaller than 0. Thus, the robot position error  $x, y$  does not diverge if the matrix  $\mathbf{A}$  from (2) has all real components of its eigenvalues negative. Eigenvalues of a  $2 \times 2$  matrix are obtainable by solving a quadratic equation

$$\lambda(\lambda + v/l) + \omega^2 = 0, \quad (3)$$

$$\lambda_{1,2} = \frac{-v/l \pm \sqrt{(v/l)^2 - 4\omega^2}}{2}. \quad (4)$$

In cases when  $(v/l)^2 < 4\omega^2$ , the expression  $\sqrt{(v/l)^2 - 4\omega^2}$  is an imaginary number, and  $\lambda_{1,2}$  are complex conjugates with the real component equal to  $-v/(2l)$ . Since  $v$  and  $l$  are always positive,  $-v/(2l)$  is always negative, which ensures the system stability and non-divergence of the robot position error  $x, y$ .

In the case of  $(v/l)^2 \geq 4\omega^2$ , the result of the square root is positive and the real part of the  $\lambda_2$  eigenvalue

$$\lambda_2 = \frac{-v/l - \sqrt{(v/l)^2 - 4\omega^2}}{2} \quad (5)$$

is always negative. The eigenvalue  $\lambda_1$ , which is calculated as

$$\lambda_1 = \frac{-v/l + \sqrt{(v/l)^2 - 4\omega^2}}{2}, \quad (6)$$

is non-negative only if  $\omega$  equals to 0. *This means, that if the robot does not move along a straight line, both eigenvalues of  $\mathbf{A}$  are lower than 0, and therefore, both longitudinal ( $x$ )*

*and lateral ( $y$ ) components of the robot position error do not diverge even if the robot uses its exteroceptive sensors only to correct its heading.*  $\square$

### B. Paths containing straight segments

According to (1) and (2), if a robot travels along a straight line, its longitudinal position error, represented by  $x$  in our model, gradually grows due to the perturbances  $s_x$ , which are caused primarily by odometric drift. Let assume that the taught path consists of straight segments conjoined by segments with non-zero curvature. Let assume that a robot started to traverse a straight segment at time  $t_0$  and ended its traversal at  $t_1$ . Its error after the segment traversal is obtainable by integrating (2) over time as:

$$\begin{pmatrix} x(t_1) \\ y(t_1) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & e^{-\frac{v}{l}(t_1-t_0)} \end{pmatrix} \begin{pmatrix} x(t_0) \\ y(t_0) \end{pmatrix} + \begin{pmatrix} b_{x0} \\ b_{y0} \end{pmatrix}. \quad (7)$$

Now, let assume that from the time  $t_1$  to the time  $t_2$ , the robot traverses a segment with non-zero curvature. Regardless of the segment shape and curvature, the magnitude of  $x$  and  $y$  decreases (see (2)), and thus we can state that

$$\begin{pmatrix} x(t_2) \\ y(t_2) \end{pmatrix} = \mathbf{N}_1 \begin{pmatrix} x(t_1) \\ y(t_1) \end{pmatrix} + \begin{pmatrix} b_{x1} \\ b_{y1} \end{pmatrix}, \quad (8)$$

where the eigenvalues of the matrix  $\mathbf{N}_1$  are lower than one. Rewriting the discrete system (7) in a compact form as  $\mathbf{x}_1 = \mathbf{N}_0\mathbf{x}_0 + \mathbf{b}_0$ , and (8) as  $\mathbf{x}_2 = \mathbf{N}_1\mathbf{x}_1 + \mathbf{b}_1$ , and substituting (7) into (8) results in:

$$\mathbf{x}_2 = \mathbf{N}_1(\mathbf{N}_0\mathbf{x}_0 + \mathbf{b}_0) + \mathbf{b}_1 = \mathbf{N}_1\mathbf{N}_0\mathbf{x}_0 + (\mathbf{N}_1\mathbf{b}_0 + \mathbf{b}_1). \quad (9)$$

Equation 9 represents a discrete system, which allows to estimate the robot position error after traversing a straight segment followed by a curved one. Since the largest eigenvalue of  $\mathbf{N}_0$  equals to 1 (see (7)) and both eigenvalues of  $\mathbf{N}_1$  are smaller than 1, the eigenvalues of their product  $\mathbf{N}_1\mathbf{N}_0$  are also smaller than 1. This means that the discrete system (9) is stable. *Thus, position error of a robot, which repeatedly traverses a path formed of conjoined straight and curved segments does not diverge even if it is using its exteroceptive sensors only to correct its heading.*  $\square$

### C. Convergence proof assumptions

The model that we established is based on two assumptions that might not be met in extreme cases. First, it assumes that during the repeat phase, the robot perceives at least some image features that it saw during the teaching. If the robot's initial position in the repeat phase is too far from the origin of the teaching step, or if the robot deviates from the path too much, the mapped features will not be in its field of view and the navigation will fail. The actual position error that would cause the navigation to fail depends on robot's camera, path shape and feature distance. In our experiments, we started the repeat phase with an initial position error exceeding 1 m, which is approximately an order of magnitude higher than the accuracy of the navigation, see Figures 5, 7, and 9. Thus, the typical navigation inaccuracy caused by  $s_x$  and  $s_y$  in (2) is very unlikely to deviate from its path beyond the point

where it can correct its position error. In practice, a robot can monitor the consistency of the feature matching and when there are not enough correspondences, it can request human intervention.

The second assumption is that the robot steering controller gain (the parameter  $\alpha$  in Section II-A) is set in a way, which allows the robot to align the mapped and currently visible features and steer itself as illustrated in Figure 1.

### III. NAVIGATION METHOD DESCRIPTION

The considered navigation system works in two steps: teach and repeat. In the teaching phase, a robot is guided by an operator along a path, which is the robot supposed to autonomously navigate in the repeat phase. During the teaching, the robot extracts salient features from its onboard camera image and stores its current travelled distance and velocity. During the autonomous navigation, the robot sets its velocity according to the travelled distance and compares the image coordinates of the currently detected and previously mapped features to correct its heading.

#### A. Image processing

The feature extraction method which detects salient objects in the onboard camera image is a critical component of the navigation system because it is the only mechanism which the robot employs to reduce its position error. Based on the results from our previous work on image feature stability in changing outdoor environments [33], we decided to use the Speeded Up Robust Features (SURF) [34] and a combination of the AGAST [35] and BRIEF [36] methods.

The feature extraction is composed of two steps, detection of keypoints and description of their vicinity. The keypoint detection indicates points in the image, which have sufficient contrast that makes them easy to localise and track. In the case of SURF, the keypoint detection is based on the approximation of Hessian matrix determinant [34], while AGAST [35] uses an optimised pixel brightness testing scheme around the keypoint candidate. To form the description of a particular keypoint, BRIEF [36] calculates a binary descriptor by comparing brightnesses of randomly-chosen pixel pairs around the keypoint. The advantage of this descriptor is an efficient calculation, low memory requirements, and rapid matching. The SURF-based descriptor is based on image intensity gradients near the keypoint [34]. While being slower to calculate and match, it is more resistant to large viewpoint changes.

Once the keypoints are detected and described, they can be matched to the keypoints stored in a map and the associations can be used to correct the robot heading. The quality of the features depends on the quality of the input image stream, which, in outdoor environments, suffers from varying illumination. To compensate for the illumination instability, we select the exposure and brightness of the robot camera based on the results from [37], [38].

#### B. Teaching (mapping) phase

During the phase, the robot is driven through the environment by a human operator. The robot continuously

measures the distance it travelled and whenever the operator changes the forward or angular velocity, the robot saves the current distance and the updated velocity values – we refer to this sequence as to a “path profile”. Additionally, the robot continuously extracts image features from its onboard camera image and every 0.2 m, it saves the currently detected image features in a local map, which is indexed by the current distance the robot travelled.

#### C. Repeat (navigation) phase

At the start of this phase, the robot loads the path profile and creates a distance-indexed list of the local maps containing the image features. Then, it sets its forward and angular velocities according to the first entry of the path profile and it loads the first local map containing data about image features visible at the start of the path. As the robot moves forwards, it extracts image features from its onboard camera image and matches them to the ones loaded from the local map. The differences of the horizontal image coordinates of the matched feature pairs (i.e., the positions of the features in the camera image relative to the positions of the features in the preloaded map) are processed by a histogram voting method. The maximum of the histogram indicates the most frequent difference in the horizontal positions of the features, which corresponds to the shift of the image that was acquired during the mapping phase relative to the image that is currently visible from the onboard camera. This difference is used to calculate a corrective steering speed, which is added to the speed from the velocity profile.

If the histogram voting results are inconclusive due to the low number of features extracted, e.g., when the robot faces a featureless wall, the camera image is over- or under-exposed, etc., the corrective angular speed is not added to the one from the velocity profile. In a case the visual information is not sufficient to determine the heading, the robot simply steers according to the path profile data. As the robot proceeds forwards along the taught path, it loads local maps and path profile data that correspond to the distance travelled and repeats the steps described above. Thus, the path profile allows the robot to steer approximately in the same way as during the teaching phase, and the image matching corrects the robot heading whenever it deviates from the intended path.

The principal advantage of the system is its robustness to feature deficiency and uneven feature distribution in the camera image. This makes the system robust to environment appearance changes and adverse lighting conditions. The histogram voting-based heading corrections are demonstrated in videos available at [32].

#### D. System implementation

The navigation system was implemented in the Robotic Operating System (ROS), in particular the version Kinetic. The system structure is shown in Figure 2. The *feature extraction* node extracts image features from the robot camera and passes them to the *mapper* and *navigator* nodes. The *odometry monitor* node receives data from robot odometry

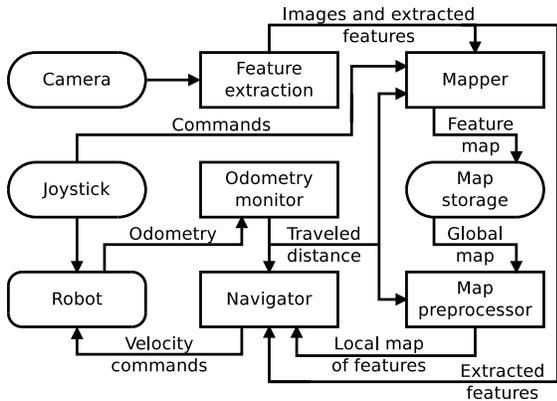


Fig. 2: Software structure of the presented system

and measures travelled distance. It also sends special messages every time the robot passes a given distance (e.g., 0.2 m), which is used by the *mapper node*, see Section III-B. The *mapper node* receives features from the *feature extraction node* and saves them into the local map when it receives the aforementioned message from the *odometry monitor node*. It also saves the path profile. The *map preprocessor node* loads all local maps and path profile, and then sends them to the *navigator node* based on the travelled distance received from the *odometry monitor*. The *navigator node* receives the velocity profile and local feature maps and it matches the features from the maps to the currently visible features from the *feature extraction node*. It performs the histogram voting described in Section III-C, calculates the robot velocities and steers it along the path. All the aforementioned modules were implemented as ROS action servers with dynamically reconfigurable parameters and are available as C++ open source code at [32].

#### IV. EXPERIMENTAL EVALUATION

The experimental evaluation of the proposed navigation system consists of 3 different experiments performed with 2 different robotic platforms. The aim of the first experiment was to demonstrate that without the visual feedback or path profile information, the robot is not able to repeat the taught path. The second experiment, which is performed under controlled conditions and an external localisation system, demonstrates the accuracy of the motion model established in Section II and the ability of a minimalistic robotic platform to converge to the taught trajectory during multiple traversals of the intended path. The third experiment demonstrates the trajectory convergence in challenging outdoor conditions including night.

##### A. Platforms

For indoor experiments, we used a lightweight robot based on an MMP-5 platform made by TheMachineLab 3. Its base dimensions are  $0.3 \times 0.3 \times 0.1$  m, its mass is 2.2 kg, and it can carry additional 2 kg payload while achieving speeds over  $1.2 \text{ ms}^{-1}$ . The robot has a four-wheel differential drive with a control unit which allows setting PWM signal duty

on individual motors by a simple serial protocol. Since the platform does not provide any odometric information, we estimate the travelled distance simply by the time and motors' PWM duty. Its vision system is based on a single



Fig. 3: MMP-5 and Cameleon ECA used in our experiments.

USB camera, which provides  $320 \times 240$  colour images with a  $45^\circ \times 35^\circ$  field of view. The computer is based on an AT3IONT-I miniATX board with an Intel Atom 330 CPU running at 1.6GHz with a 2GB RAM. Due to its computational constraints, this robot is using the AGAST/BRIEF features.

For outdoor experiments, we used a heavy duty, tracked platform, called CAMELEON ECA, which is equipped with onboard PC and two cameras as primary sensors. Unfortunately, the cameras are positioned very low, which causes problems in grassy terrains and the onboard PC is too slow. Thus, we equipped the robot with a superstructure with several equipment mounts, where we placed the TARA USB stereo camera (we use only the left camera image in our experiments), Intel i3 laptop with 8GB RAM and the Fenix 4000 lumen torch, see Figure 3.

##### B. Experiment I: Proof-of-concept

To evaluate the system's ability to repeat the taught path and to correct position errors that might arise during the navigation, we have taught the CAMELEON platform a closed, approximately 25 m long path in the outdoor environment in the Czech Technical University campus. The shape of the trajectory is a closed, smooth, oval-shaped curve. After mapping, we let the robot to drive along the taught path repeatedly. Every time the robot completed a path loop, we measured its distance relative to the path end/start. In this way, we quantitatively assess the robot's ability to adhere to the path it has been taught. The experiments were performed during the evening, and therefore, the lighting conditions changed from high to low illumination, which made the image-based navigation particularly difficult. Facing changes in environment and lighting conditions is inevitable for long-term navigation, that is why we chose this particular setup.

To demonstrate the interplay between the path profile velocity setting and vision-based heading correction, we let the robot to drive the path autonomously using only the velocities remembered from the teaching phase (i.e. path profile), then only using visual information and then the combination of these. In the first test, we have deactivated the vision-based heading corrections and we let the robot

move according to the path profile only – this corresponds to the term  $vyl^{-1}$  in (2) being equal to 0. The model (2) predicts that the errors of  $s_x$  and  $s_y$  will gradually accumulate, drifting the robot off the taught path. As predicted by the model, during this trial the robot slowly (by  $\sim 1$  m every loop) diverged from the taught path because of the inaccurate odometry.

During the second trial, we have let the robot run with the method described in [30]. Thus, the robot did not use the path profile information, but it moved forward with a constant speed and steered its heading according to the results of the image feature matching and histogram voting. In this case, the robot diverged from the taught path as soon as it was supposed to perform a sharper turn, because the visual heading correction by itself could not perform sharp turns and the mapped features were lost from robot’s field of view.

The final trial used both path profile and visual feedback as described in Section III. To verify if the robot can correct position errors that arise during navigation, we have started the autonomous navigation, not at the path start, but 0.9 m away in longitudinal and 0.8 m away in lateral direction. The reason for the robot displacement is to demonstrate that unlike in the previous two cases, where the position error diverged, a combination of the vision and path profile information would allow the robot to suppress the error and adhere to the taught trajectory. As expected, each time the robot completed the taught path, its position error decreased, which confirms the assumptions stated in Section II. Further details about this experiment are provided in a short paper [39].

### C. Experiment II: Position error model verification

The indoor experiments were meant to compare the real system behaviour with the model of the robot movement. In these experimental trials, we used the MMP-5 robot platform equipped with a circular marker, which allows for an accurate tracking of the robot position. In the first trial, we guided the robot along a 10 m long oval-shaped path consisting of two half-circle segments connected with two straight lines, see Figure 4. Then, we displaced the robot 1 m away from the path start and let it traverse along the path autonomously 20 times while recording its position with an external localisation system [40], [41].

Figure 4 shows the robot trajectory during the autonomous traversals, which slowly converges to the mapped path. Each time the robot finished one path traversal, we measured its position relative to the path start. Fig. 5 shows that the position error diminished after two loops, stabilising at 7 cm.

The convergence of the robot position during the autonomous travels is visible in Figure 4 and the robot position error evolution along the first path traversal in Figure 5. The error evolution confirms the mathematical model presented in Section II – one can observe that during the first 5 meters, where the path is curved, the position error diminished rapidly. Once the robot starts to traverse the straight segment, the position error stabilises and it starts to diminish once

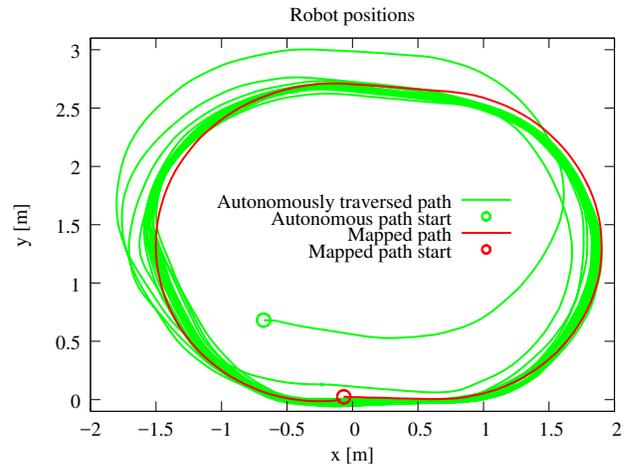


Fig. 4: Indoor trial I: Robot path during teach and repeat.

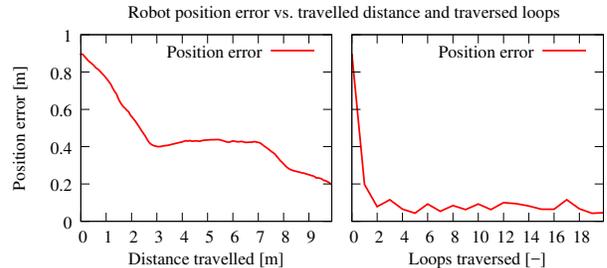


Fig. 5: Indoor trial I: Robot position error during the first autonomous path traversal (left) and during the 20 autonomous path repeats (right).

again when the robot starts to traverse the second semi-circular path segment. Since the robot is nonholonomic, skid-steer drive, the convergence of the position error implies that its orientation conforms to the model (1).

In the second trial, we guided the robot along a 17 m long lemniscate-shaped path consisting of four half-circle segments connected with three straight ones and let it traverse the path 19 times, see Figure 6. This trial was performed in a larger hall than the previous one, and therefore, the average distance of landmarks is bigger than in the previous case, causing slower convergence of the robot trajectory, see Figure 7. Furthermore, the start and end of the taught path are about 0.15 m apart, which causes the robot to start with 0.15 m error during subsequent path traversals, which is notable in Figure 6, where the first part of the repeated path is slightly displaced from the taught one. Similarly to the previous case, the error evolution shown in the left part of Figure 6 conforms with the mathematical model presented in Section II – the error reduction is slower during traversal of the straight segments of the taught path. In these experiments, a robot without any odometric system, equipped with a low resolution, uncalibrated camera which suffered from motion blur (see [32]) reliably traversed over 700 m, reducing the initial  $\sim 1$  m position errors below  $\sim 0.1$  m.

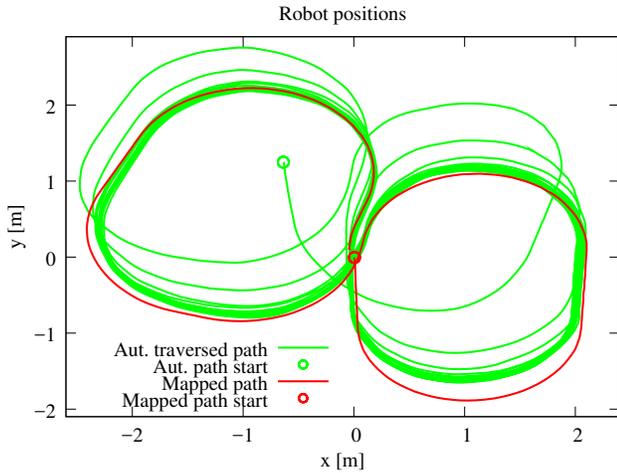


Fig. 6: Indoor trial II: Robot path during teach and repeat.

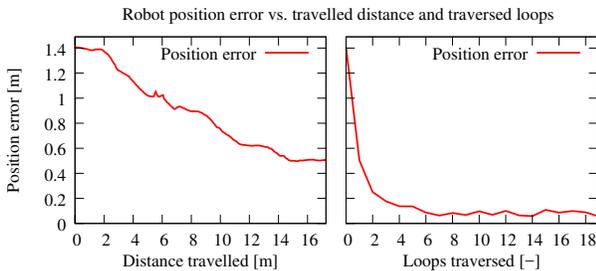


Fig. 7: Indoor trial II: Robot position error during the first autonomous path traversal (left) and during the 19 autonomous path repeats (right).

#### D. Experiment III: System robustness

The purpose of final outdoor experiments is to demonstrate the ability of the system to cope with variable outdoor illumination. The first trial was performed during a day, where a robot was supposed to traverse a 60 m long path at the Hostibejk hill in Kralupy nad Vltavou, see Figures 3 and 8. The second and third trials were performed at night at the same location. During the first trial, we created the map in a clear sky weather and let the robot traverse the path 7 times one month later during a cloudy day.

In the second trial, which was performed at night, we attached a 4000 lumen searchlight to the robot’s superstructure. The location of the trial was free of any artificial light sources; so, the most of the robot path, it saw only parts of the scene, which it illuminated itself, see Figure 8. After creating the local map, we displaced the robot by 1.5 m and let it traverse the path 7 times (the number of traversals is limited by the capacity of the searchlight batteries).

The third trial was performed at the same location, which at this time was partially illuminated, because three of the local street lamps were repaired in the meantime. Again, after teaching the robot a 60 m long path, we displaced the robot by 1.5 m and let it traverse the path 7 times. During these trials, we initiated the autonomous run 1.5 m from the



Fig. 8: Outdoor experiment: Hostibejk site at night and day from the robot perspective.

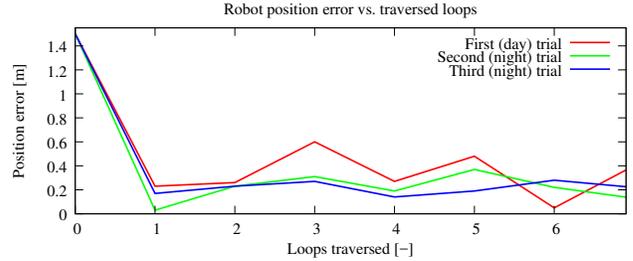


Fig. 9: Outdoor experiments: Robot position error relatively to the path start after traversing the path  $n$  times.

taught path start and then measured the robot displacement from the path start every time it completed the taught path. Figure 9 shows that the initial position error quickly diminished to values around 0.2–0.3 m. In these experiments, a tracked robot with imperfect odometry, equipped with an uncalibrated camera working in low-visibility conditions (see [32]) reliably traversed over 1.2 km, reducing the initial  $\sim 1.5$  m position errors to  $\sim 0.2$ –0.3 m. Furthermore, we did 10 days of additional tests at the same site over a period of one month. In these tests, the robot successfully traversed the taught path 66 times, 21 during the day, 25 during the night and 20 during the sunset and covered distance over 4 km.

To compare the system’s robustness to the state-of-the-art SLAM-based methods, we processed the data gathered during these trials by the ORB-SLAM [8], which we modified to take into account odometric information when initialising camera position estimation. To do so, we had to calibrate the camera and process the gathered data (in the form of ROSbags) by the ORB-SLAM. In our tests, we had to replay the ROSbags from the day trial  $2.5\times$  slower than in the original experiment several times because of occasional loss of tracking and subsequent breakdown of the ORB-SLAM method. Despite trying several settings of ORB-SLAM, we could not build a consistent map from the night data due to feature deficiency and motion blur. This comparison indicates the proposed method’s robustness to difficult illumination conditions.

#### V. CONCLUSION

We formulated a mathematical proof which indicates that in teach-and-repeat scenarios, explicit localisation of a mobile robot along the taught route is not necessary. Rather, a robot navigating through a known environment can use an environment map and visual feed to correct only its heading,

while measuring the travelled distance only by its odometry. Based on this principle, we designed and implemented a simple teach-and-repeat navigation system and evaluated its performance in a series of experimental trials. These experiments confirmed the validity of the aforementioned proof, showing that this kind of simple navigation is sufficient to keep the position error of the robot limited. The requirement to establish only the robot heading simplifies the visual processing and makes it particularly robust to situations, where the detected and mapped features cannot be associated reliably. This makes our system capable of handling difficult lighting conditions and environmental changes, which is demonstrated in several experiments. Compared to the previous work [30], the mathematical proof of bearing-only navigation presented here is simpler, shorter and is not limited to polygonal routes only. Thus, unlike in [30], where the robot could only learn polygonal routes in a turn move manner, our navigation system allows to learn arbitrarily-shaped routes, making its real-world deployment more feasible. Furthermore, we show that the robot position error is asymptotically stable, whereas in our previous work [30] we only proved its Lyapunov stability. In our latest work, we extended the system so that it's able to improve its performance by exploiting the experience gathered during autonomous traversals [42], [38], [43].

#### REFERENCES

- [1] G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2002.
- [2] F. Bonin-Font, A. Ortiz, and G. Oliver, "Visual navigation for mobile robots: A survey," *Journal of intelligent and robotic systems*, 2008.
- [3] J. Alvarez and A. Lopez, "Road detection based on illuminant invariance," *IEEE Trans. on Int. Transportation Systems*, 2011.
- [4] B. Wang, V. Frémont, and S. A. Rodríguez, "Color-based road detection and its evaluation on the KITTI road benchmark," in *Intelligent Vehicles Symposium*, 2014.
- [5] B. Åstrand and A.-J. Baerveldt, "A vision based row-following system for agricultural field machinery," *Mechatronics*, 2005.
- [6] A. Kosaka and A. C. Kak, "Fast vision-guided mobile robot navigation using model-based reasoning and prediction of uncertainties," *CVGIP: Image understanding*, vol. 56, no. 3, pp. 271–329, 1992.
- [7] S. Holmes, G. Klein, and D. W. Murray, "A Square Root Unscented Kalman Filter for visual monoSLAM," in *International Conference on Robotics and Automation (ICRA)*, 2008, pp. 3710–3716.
- [8] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.
- [9] 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*, 2017.
- [10] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular slam," in *Eur. Conf. on Computer Vision*, 2014.
- [11] R. Wang, M. Schwörer, and D. Cremers, "Stereo dso: Large-scale direct sparse visual odometry with stereo cameras," in *International Conference on Computer Vision (ICCV)*, Venice, Italy, October 2017.
- [12] C. Cadena *et al.*, "Past, present, and future of simultaneous localization and mapping: Towards the robust-perception age," *IEEE Transactions on Robotics*, 2016.
- [13] K. Kidono, J. Miura, and Y. Shirai, "Autonomous visual navigation of a mobile robot using a human-guided experience," in *International Conference on Intelligent Autonomous Systems*, 2000.
- [14] P. Furgale and T. D. Barfoot, "Visual teach and repeat for long-range rover autonomy," *Journal of Field Robotics*, vol. 27, no. 5, pp. 534–560, 2010. [Online]. Available: <http://dx.doi.org/10.1002/rob.20342>
- [15] E. Royer, M. Lhuillier, M. Dhome, and J.-M. Lavest, "Monocular vision for mobile robot localization and autonomous navigation," *International Journal of Computer Vision*, 2007.
- [16] W. S. Churchill and P. Newman, "Experience-based navigation for long-term localisation," *IJRR*, 2013.
- [17] F. Dayoub, G. Cielniak, and T. Duckett, "Long-term experiments with an adaptive spherical view representation for navigation in changing environments," *Robotics and Autonomous Systems*, 2011.
- [18] G. D. Finlayson, S. D. Hordley, C. Lu, and M. S. Drew, "On the removal of shadows from images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 1, pp. 59–68, Jan 2006.
- [19] 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 *IROS*, 2016.
- [20] M. Paton, K. MacTavish, C. J. Ostafew, and T. D. Barfoot, "It's not easy seeing green: Lighting-resistant stereo visual teach & repeat using color-constant images," in *ICRA*, 2015.
- [21] K. MacTavish, M. Paton, and T. D. Barfoot, "Visual triage: A bag-of-words experience selector for long-term visual route following," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2017.
- [22] M. Paton, F. Pomerleau, and T. D. Barfoot, "In the dead of winter: Challenging vision-based path following in extreme conditions," in *Field and Service Robotics*, 2016.
- [23] N. Hawes *et al.*, "The strands project: Long-term autonomy in everyday environments," *IEEE Robotics and Automation Magazine*, 2016.
- [24] N. J. Cowan, J. D. Weingarten, and D. E. Koditschek, "Visual servoing via navigation functions," *IEEE Transactions on Robotics and Automation*, Aug 2002.
- [25] G. Blanc, Y. Mezouar, and P. Martinet, "Indoor navigation of a wheeled mobile robot along visual routes," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2005.
- [26] Y. Matsumoto, M. Inaba, and H. Inoue, "Visual navigation using view-sequenced route representation," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Minneapolis, USA, 1996.
- [27] S. Segvic, A. Remazeilles, A. Diosi, and F. Chaumette, "Large scale vision based navigation without an accurate global reconstruction," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.
- [28] Z. Chen and S. T. Birchfield, "Qualitative vision-based path following," *IEEE Transactions on Robotics and Automation*, 2009.
- [29] T. Nguyen, G. K. I. Mann, R. G. Gosine, and A. Vardy, "Appearance-based visual-teach-and-repeat navigation technique for micro aerial vehicle," *Journal of Intelligent & Robotic Systems*, 2016.
- [30] T. Krajník, J. Faigl, V. Vonásek *et al.*, "Simple, yet Stable Bearing-only Navigation," *Journal of Field Robotics*, 2010.
- [31] T. Krajník, S. Pedre, and L. Přeučil, "Monocular navigation for long-term autonomy," in *Int. Conf. on Advanced Robotics (ICAR)*, 2013.
- [32] F. Majer, L. Halodová, and T. Krajník, "Source codes: Bearing-only navigation." [Online]. Available: [https://github.com/gestom/stroll\\_beamnav](https://github.com/gestom/stroll_beamnav)
- [33] T. Krajník, P. Cristóforis, K. Kusumam, P. Neubert, and T. Duckett, "Image features for visual teach-and-repeat navigation in changing environments," *Robotics and Autonomous Systems*, 2017.
- [34] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, 2008.
- [35] E. Mair, G. D. Hager, D. Burschka, M. Suppa, and G. Hirzinger, "Adaptive and generic corner detection based on the accelerated segment test," in *European conference on Computer vision*, 2010.
- [36] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: binary robust independent elementary features," in *Proceedings of the ICCV*, 2010.
- [37] L. Halodová and T. Krajník, "Exposure setting for visual navigation of mobile robots," in *Student Conf. on Planning in AI and Robotics (PAIR)*, 2017.
- [38] L. Halodová *et al.*, "Adaptive image processing methods for outdoor autonomous vehicles," in *Modelling and Simulation for Autonomous Systems (MESAS)*, 2018, to appear.
- [39] F. Majer, L. Halodová, and T. Krajník, "A precise teach and repeat visual navigation system based on the convergence theorem," in *Student Conf. on Planning in AI and Robotics (PAIR)*, 2017.
- [40] T. Krajník, M. Nitsche, J. Faigl, P. Vaněk, M. Saska, L. Přeučil, T. Duckett, and M. Mejail, "A practical multirobot localization system," *Journal of Intelligent and Robotic Systems*, 2014.
- [41] F. Arvin, T. Krajník, A. E. Turgut, and S. Yue, "COSΦ: artificial pheromone system for robotic swarms research," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.
- [42] E. Dvořáková, "Temporal models for mobile robot visual navigation," Bachelor thesis, Czech Technical University, May 2018.
- [43] L. Halodová, "Map management for long-term navigation of mobile robots," Bachelor thesis, Czech Technical University, May 2018.