Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Robot Navigation on Slow Feature Gradients

Muhammad Haris, Mathias Franzius, Ute Bauer-Wersing

2018

Preprint:

This is an accepted article published in International Conference on Neural Information Processing (ICONIP 2018). The final authenticated version is available online at: https://doi.org/10.1007/978

Robot Navigation on Slow Feature Gradients

Muhammad Haris[2] and Mathias Franzius[2] and Ute Bauer-Wersing[1]

[1] Faculty of Computer Science and Engineering, Frankfurt University of Applied Sciences, 60318 Frankfurt, Germany
[2] Honda Research Institute Europe GmbH, 63073 Offenbach, Germany

Abstract. Unsupervised learning with Slow Feature Analysis (SFA) enables an agent to learn spatial representations of its environment from images captured during an exploration phase. In a subsequent application phase, slow features encode the robot's position. The representation is spatially smooth and implicitly encodes the average travel time during exploration. Following the SFA gradient allows the robot to navigate even around obstacles without any planning. Earlier work showed this basic principle in noise-free simulation, using two virtual cameras on a robot. We extend the approach to be more robust and more computationally efficient. We test it on a lawn mower robot with a single camera for navigation in free space and avoiding obstacles.

1 Introduction

One of the most important abilities for autonomous robots is navigation in space. For nontrivial navigation, a mobile agent requires representations of the environment and its location to plan a viable path to a target. Visual simultaneous localization and mapping (vSLAM) is the state of the art approach for generating a map of the environment and simultaneous position estimation based on camera data. Maps generated by SLAM can take various forms, e.g. topological or as occupancy grids, which form the basis for a variety of navigation algorithms [1]. Such navigation approaches fall into a range of computational complexity from simple reactive motion approaches to trajectory planning in metrical maps [2]. Navigation is still a challenging task for mobile agents, whereas rodents, for example, are excellent at navigation. The resulting paths may not be optimal, but very flexibly and quickly generated, leading to an adaptive and robust navigation behavior. One such model is RatSLAM [3]. The agent's position and orientation are encoded as an activity packet in a continuous attractor network. Odometry and image matching introduce energy, shifting the activity peak. A later version of the model [4] allows it to keep spatial representations consistent for longer time periods in a graph-like experience map. Another class of models is based on slowness learning [5, 6] and also related to localization as a model of the rodent hippocampus. In slowness learning [7], the resulting representation can achieve significant invariances, e.g., to head orientation. Using an uncalibrated catadioptric imaging system, an earlier model successfully allowed a mobile robot to localize in an outdoor environment [8]. The learned representations of such

models can be projected into metric space for navigation. While such a step makes quantitative evaluation easier and allows, for example, the integration of Kalman Filtering, it requires additional computational effort and reduces the richness of the representation.
Navigation in topological maps is an established method and can be efficiently done with the graph search algorithm A* [9]. While this algorithm provably finds an optimal path in relevant settings, it can scale unfavorably in large environments with many obstacles. A different navigation approach for metric spaces is based on force fields generated by an attractor at the target and repulsive forces from obstacles [10, 11]. While this potential field approach is elegant and simple, it is strongly limited by local minima, e.g. for navigation around non-convex obstacles [12]. This restriction does not apply to the proposed method, as we will show.
Navigation by reinforcement learning based on slow features from visual data has been shown before [13, 14]. However, these approaches require extensive additional learning for new target positions.
We present results for robot navigation on the slow feature gradients. The features are orientation-invariant and generated with the model presented in [8] and [15]. After the learning phase, navigation is performed by following the feature gradients approximated from cost measurements at the target and the current position. The previous work [15] was limited to simulation and required multiple cameras for estimating the slow feature gradients. Here, we extend the approach to become practical in a number of ways. First, we improve robustness, computation speed, and orientation invariance by replacing a part of the Slow Feature hierarchy with a Fourier feature extraction as preprocessing. Second, we present an approach to estimate the navigation gradient with a single camera. Finally, we present quantitative results for navigation with a lawn mower robot in free space and around obstacles.

2 Learning Spatial Representations with SFA

2.1 Slow Feature Analysis

Slow Feature Analysis transforms a set of time varying input signals $\boldsymbol{x}(t)$ into a set of slowly varying output signals $\boldsymbol{y}(t)$. The optimization objective is to find functions $g_j(\boldsymbol{x})$ such that the output signals [7]

$$y_j(t) := g_j(\boldsymbol{x}(t))$$

minimize

$$\Delta(y_j) := \langle \dot{y}_j^2 \rangle_t$$

under the constraints

$$\begin{aligned} \langle y_j \rangle_t &= 0 \text{ (zero mean)}, \\ \langle y_j^2 \rangle_t &= 1 \text{ (unit variance)}, \\ \forall i < j : \langle y_i y_j \rangle_t &= 0 \text{ (decorrelation and order)} \end{aligned}$$

The $\langle \cdot \rangle_t$ and $\dot{y}$ represents time averaging and derivative of y, respectively. The Δ-value defines the temporal variation of the output signal $y_j(t)$ and its minimization is the objective function. A lower value indicates less variation of the signal over time thus means slowly varying signals. The first two constraints normalize all output signals to make their time derivative comparable and avoid trivial solution, $y_j(t) = constant$. The last constraint enforces that different output signals code for distinct features of the input. We employ the MDP implementation of the SFA algorithm [16].

2.2 Learning Spatial Representations

The navigation task requires spatial representations that should code for the position of the robot in space and are independent w.r.t. its orientation. While the objective of SFA enforces temporal slowness on the training trajectory, a suitable intermeshed training trajectory leads to representations that also change smoothly in space [6]. Furthermore, if the environment is suitably rich (i.e., without an identical appearance at different locations) and the function space for training is sufficiently large, each position in space is uniquely encoded by its Slow Feature representation[3]. Together, these properties allow a robot to navigate by following the gradient of the difference between the spatial representations of the current position and a target position. The encoding of position or orientation in the learned slow feature representations depends on the movement statistics of a robot during the training run. If the robot's orientation changes sufficiently faster than its position, the slowest features become orientation invariant [6]. In earlier work, we simulated robot rotation artificially by rotating the images from a 360-degree camera on the robot. [8, 15]. Learning orientation invariance, however, generates computational load during training and the results are not perfect for limited training data. Noise on Slow Feature representations deteriorates the navigation gradient and is thus more problematic for gradient based navigation than for localization. As a consequence, we designed the current system with a preprocessing step that explicitly removes orientation dependency from the input representations for SFA. For this purpose, the omnidirectional images captured during the training phase are projected to panoramic images. Each panoramic image consists of 30 rows, which are expanded into Fourier series and only the magnitude part corresponding to the lowest 15 frequency components are stored. This provides a way to achieve rotation invariance as

[3] Please note that in contrast to PCA, SFA extracts spatial representations cf. [7].

magnitude coefficients of Fourier components are not dependent on robot direction [17, 18].
Fourier components obtained for each image by the preprocessing step are used to learn SFA representations. The learning phase has two steps, the first step reduces the dimensionality using a linear SFA while the second step extracts non-linear slow features using a quadratic SFA. The input and output dimensionality for the first step are 450 and 20, respectively. The input and output dimensionality for the second step are 20 and 8, respectively.
We can visualize the SFA representation with so-called spatial firing maps. These maps represent the color-coded function values for each position in a training trajectory. Theory has shown that for localization in open space with a relatively high rotational speed the first two SFA-outputs are orthogonal and monotonic [6], so in this case, there is one global minimum for navigation. The features depicted in Fig. 1 clearly encodes spatial information, with a gradient along the coordinate axes of the two slowest features. In contrast to the optimal solutions in open space as in [6], the representations are distorted around the obstacle. Due to the slowness objective of SFA, views of temporally nearby data during training are encoded with similar slow feature values, while temporally distant data is encoded with distinct values due to the unit variance constraint of SFA. This implicitly encodes the information about obstacles present in a training area as the slow feature values on either side of an obstacle will be highly different. Fig. 1 is based on simulation data. Unfortunately, we can not provide such plots for the actual robot scenario, because ground truth data are hard to obtain in our setting and the metric positions we recorded during the experiments are of insufficient quality. Please note, that our approach presented here is unsupervised, hence ground truth data mostly serves the purpose of metric visualization and metric evaluation.

3 Navigation Method

We use slow features gradients to navigate between arbitrary points in a two-dimensional space. We assume that the target point's slow feature representation is known (e.g. stored during training). We define a cost function C as the Euclidean distance between slow feature representations of points in a 2D space. The approximate gradient of the cost surface determines the navigation direction, thus it is possible to implement navigation by performing gradient descent on the cost surface C. For an n-dimensional slow feature space, the mapping function $f : \mathbb{R}^2 \mapsto \mathbb{R}^n$ maps a position to the slow feature space by processing the associated image. We use Euclidean distance as a cost function $C : \mathbb{R}^n \mapsto \mathbb{R}$, which calculates the cost from current position $p := (x_p, y_p)$ to target position $t := (x_t, y_t)$ with input $f(p)$ [15]:

$$C(f(p)) = \sqrt{\sum_{i=1}^{n} (f(p)_i - f(t)_i)^2}$$

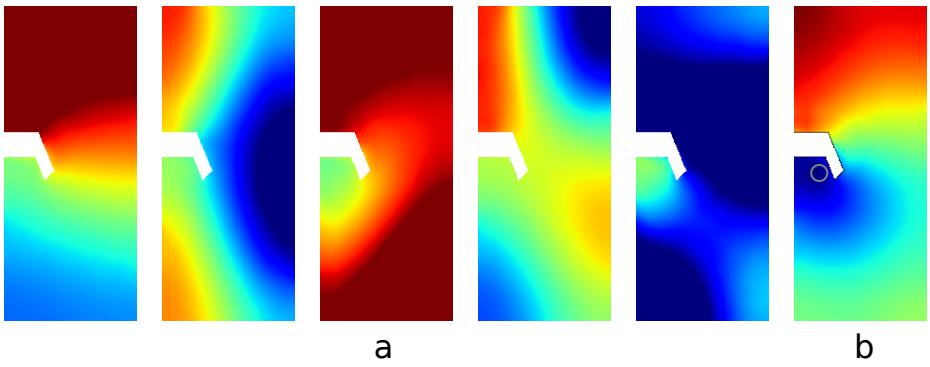


Fig. 1. Simulated spatial firing maps: (a) Spatial firing maps of the first five SFA units $y_{1..5}$. The maps of the first two SFA units $y_{1,2}$ together uniquely and smoothly encode the robot's position. The maps of functions $y_{3..5}$ show a mixture of the first two units and/or higher modes. (b) The plot shows an example cost surface in a slow feature space when the target position is in the convex region of the obstacle as indicated by the circle. Performing gradient descent on this surface allows a robot to circumnavigate the obstacle.

However, we compute a local linear approximation of the gradient as the analytical gradient $\frac{\partial C(f(p))}{\partial p}$ is infeasible to obtain. Here, we use the robotic odometry for estimating the metric position of previous observations. Note that odometry for the recent past is very precise but deteriorates quickly over long distances.

3.1 Implementation

The estimation of the gradient direction at any given position requires the cost measurement at a current and at least two nearby points. These points should not be collinear to fit a plane to a cost surface. In contrast to previous work [15], where two measurements were available at each distinct position, in our current setting, only a single omnidirectional camera is mounted on a mobile robot. Therefore, it is not possible to obtain a gradient estimation directly from the starting point of navigation. For this reason, the robot traverses a fixed V-shaped initial trajectory and captures all the images during traversal. The next step is to compute the cost at each position by transforming all the captured images in the slow feature space. Using the robot's odometry information, an estimation matrix is then created that contains the coordinates and the associated cost value (x, y, C) for each visited position. This is followed by the application of Singular Value Decomposition (SVD) on the estimation matrix to compute the gradient direction. Please note that although three non-collinear points are sufficient to estimate a gradient at any given position p, we use all the intermediate points to increase the robustness of gradient estimation. The constructed estimation matrix for each segment on a navigation path is also retained as a history of past values to use some of them for estimating future gradients depending on

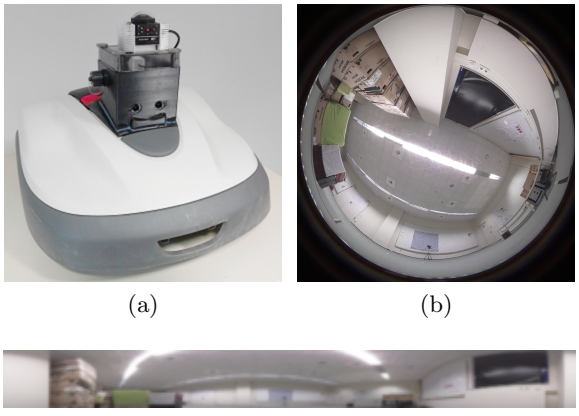
the size of the window. The decision of choosing the window size depends on the experiment's scenario (i.e., navigation in an open field or around an obstacle). We parametrize the window size by keeping the full history for free area navigation and a subset of it for obstacle circumnavigation. Using a smaller window size for the latter case allows the robot to take sharp turns–otherwise, it would result in averaging out the gradients.
The odometry quality degrades over time, however, it is still good for local position estimation which the robot uses for motor control. After gradient estimation, the robot moves along the estimated gradient and also captures all the intermediate images. A cost value threshold in slow feature space and a maximum number of iterations serve as stopping criteria. The process is repeated until one of the conditions is met. Each estimated gradient is normalized and multiplied with a constant scaling factor η. Moreover, we used a momentum term γ, which includes information from previous gradients to improve convergence [15]. The navigation algorithm given below uses a previously learned slow feature representation f and a known target representation. (See Algorithm 1).

1 Navigation algorithm

$\gamma = 0.3$ ▷ momentum term
$\eta = 0.5$ ▷ gradient scaling factor
previous_gradient = None
Traverse a fixed V-shaped trajectory of length 1.96 m and capture images I
do
    for each image i in I **do**
        k = project omnidirectional image i to a panoramic image
        p = row-wise Fourier expansion of k
        c = compute cost $C(f(p))$
        x,y = retrieve camera position from odometry information
        append (x,y,c) to estimation matrix E
    end for
    g = apply SVD on E to compute direction of steepest descent
    normalized_gradient = normalize g
    if previous_gradient is not None **then**
        gradient = γ * previous_gradient - η * normalized_gradient
    else
        gradient = η * normalized_gradient
    end if
    previous_gradient = gradient
    Move along the gradient direction and capture images I
    c = compute cost $C(f(p))$ at the updated position
while c < 0.3 or number_of_iterations == 200

4 Experiments

Experiments were performed in an indoor environment having a training area of size 4×10 meter. A V-shaped obstacle was placed inside the training area to later test the navigation performance around obstacles. We used a modified autonomous lawn mower equipped with a single omnidirectional camera for the experiments. The omnidirectional camera recorded images with a resolution of 2880×2880 pixels, which are projected to panoramic images with a resolution of 300×30 pixels. Fig. 2 shows the robot, an omnidirectional image, and the associated panoramic image. For the training phase, we put the lawn mower in a free mowing mode and let it run for approximately 50 minutes and captured $15,000$ images to assure that it covers the environment uniformly. In this mode, it drives straight segments until it detects the border, where it turns into a random direction. An example training trajectory is shown in Fig. 3. After the completion of the training phase, slow features were computed directly from Fourier components of a single image. We only used the first two slow feature unit values $s_{1,2}$ for all our experiments as in [15]. The gradient scaling factor and the momentum are set to $\eta = 0.5$ and $\gamma = 0.3$, respectively for all experiments. The maximum number of iterations and cost value threshold in slow feature space are set to 200 and 0.3 as stopping criteria, respectively.



(a) (b)

(c)

Fig. 2. Robot and Real-world environment. (a) A modified lawn mower robot with an omnidirectional camera is used for experiments. (b) An omnidirectional image. (c) The corresponding panoramic image.

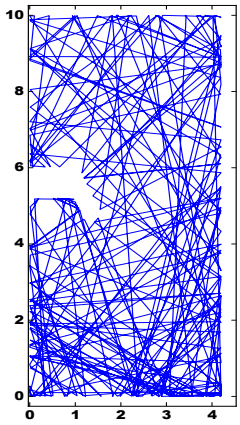


Fig. 3. An example training trajectory, which has straight line segments with random orientation. A V-shaped obstacle is present in the field.

4.1 Navigation in a Free Space Scenario

We start with free area navigation. For this case, we chose start and end positions such that there is no obstacle on the shortest path between them. We tested our approach for five different start and end positions with five trials each.

Results Fig. 4 shows the results of four different start and end positions. We evaluated the goal approach rate as a performance measure. The robot always reached within the close vicinity of target location but it stops at different distances w.r.t. target location even for the trials of same start and end positions. We varied the success criterion which is the final distance to target location between 0 and 1 meter and plotted the goal approach rate for all test runs. It is shown in Fig. 5. The goal approach rate varies between 0% to 96%. For a target to distance of 0.8 meters, for example, the goal approach rate is 85%.

4.2 Navigation around an Obstacle

For the second scenario, we tested our approach to validate that the slow feature gradients allow a robot to navigate around obstacles without the need of explicit planning. For these experiments, the target location was kept fixed inside the convex region of the obstacle while different start positions are chosen from the opposite side of the obstacle. We performed experiments for four different starting positions with five trials each.

Results The resulting 20 trajectories are shown in Fig. 6. Here, we repeated the same analysis as for the first scenario. The goal approach rate varies between 0% to 75%. For a success criterion of 0.8 meters, the goal approach rate is 70%. The plot for goal approach rate is shown in Fig. 5. The performance decrease

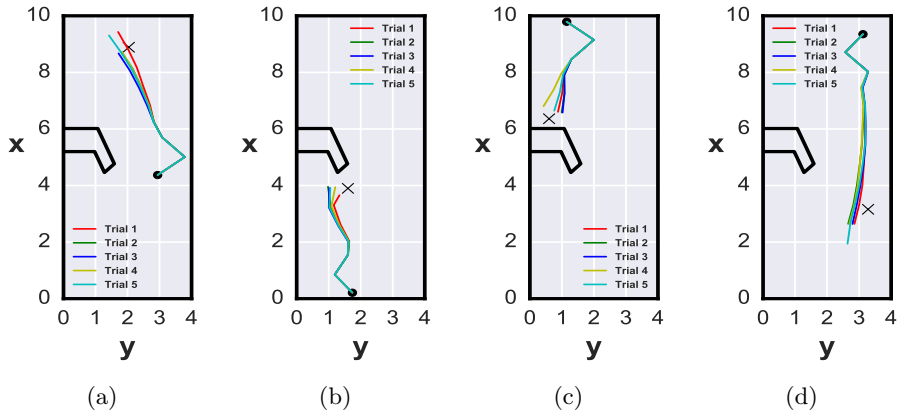


Fig. 4. Navigation in free space. The start and target positions are marked by a dot and cross, respectively. (a)-(d) Results of four different start and end positions using the first two slow feature units. A successful trial is the one that has a final distance of less than 0.8 meters from a target location.

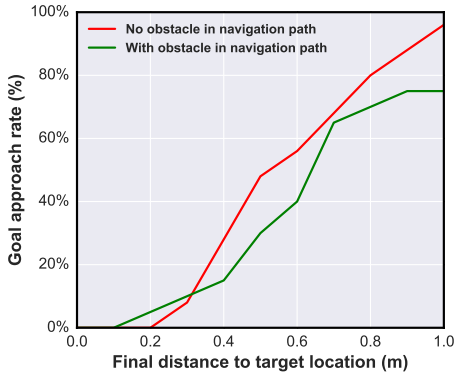


Fig. 5. Goal approach rate: The red line shows the performance tradeoff for free space navigation while the green line shows the performance tradeoff for obstacle circumnavigation. For the criterion of 0.8 meter as the final distance to the target location, the robot reached the target location 85% of the trials for free space and 70% of the trials for circumnavigation.

comes from the cases of the robot navigating very closely and colliding with the obstacle. In these cases, the robot hit the obstacle and never navigated to the target location. We encountered four such cases out of a total of twenty test runs. For our current analysis, we treated those cases as a failure. Another reason for failed navigation is that the robot got stuck in local minima due to flat gradients as the most variance is concentrated in the regions near to obstacle (see Fig. 1 (b)).

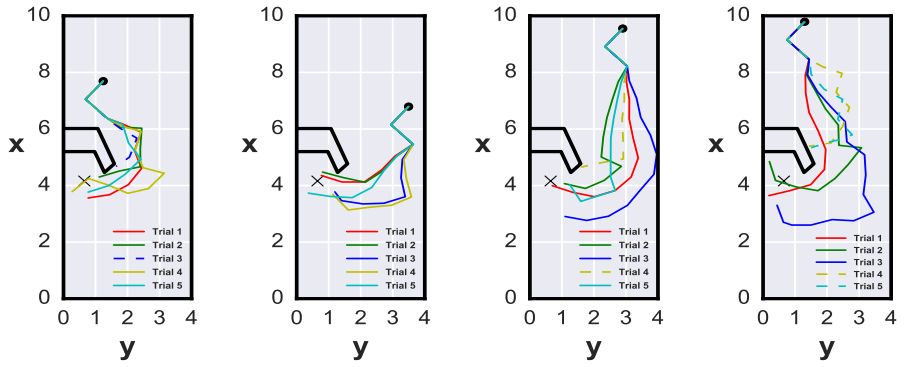


Fig. 6. Resulting trajectories for navigation around obstacles. The start and target positions are marked by a dot and a cross, respectively. (a)-(d) Four different start positions with five trials each. The robot navigated around the obstacle using the first two slow feature unit values. The trajectories for the cases where the robot hit the obstacle are indicated by dashed lines. A successful trial is the one that has a final distance of less than 0.8 meters from a target location.

5 Conclusion

We presented a robot navigation system that works in slow feature space by applying gradient descent to a cost surface. The system learned spatial representations for the specific environment during an offline training phase where the robot follows the standard movement pattern of a robotic lawnmower. After the unsupervised learning step, the robot can navigate by following the gradient of the difference between its current spatial representation and the target representation. The direction of steepest descent is estimated from images in the recently traveled past. Locations that are not separated by obstacles were typically visited temporally close during training and are thus encoded with similar Slow Feature values. Views from different sides of an obstacle, on the other hand, were never seen in temporal proximity and get more different representations. Thus, the resulting slow feature representations implicitly encode average travel time during training (e.g., obstacles). Hence, navigating around obstacles requires no specific planning of the trajectory but is achieved by following the steepest gradient. The real robot reached the target in 85% of the trials in an open field scenario and in 70% of the trials when the target was behind the obstacle. Hence, we were able to reproduce the navigation results of extensive statistical experiments from noise-free simulator environment [15]. The variability in each set of trajectories is probably due to small variances in start pose and environmental changes. In some failure cases the robot got stuck in regions with flat gradients. Due to the presence of the obstacle in the training area, the resulting representations distort around it, which leads to relatively

flat gradients for most regions of the training area as also depicted in the simulated experiments [15]. An advanced gradient descent algorithm may help with such issues. Further, an extension to the SFA-algorithm made in [19] presents a feasible solution to overcome this issue. Other failures occurred when the robot hit the obstacle. In the future, the robot may fall back to standard behavior in such a case, drive a short distance into a random direction and then resume navigation. Alternatively, an obstacle avoidance sensor on the robot may be used to keep a minimum distance to obstacles. Environmental changes may directly influence the spatial representations, however, we can improve them by using the strategies presented in [20, 21].

The approach is not limited to obstacle avoidance by replicating the movement statistics of a training phase. If, for example, the training data is collected from human remote operation, autonomous navigation would replicate the operator's preferences for path planning.

The computational load for our proposed system is orders of magnitudes lower than of standard SLAM or Deep Learning systems, as well as those based on Slow Feature Hierarchies. Learning and application phases are suitable for small and cheap systems based on digital signal processors (DSPs). Our model is especially suited for an application where the robot has a default operation mode with random navigation during which the SFA learning can occur. Over time, the robot's efficiency and capabilities can then improve as described.

References

1. Fuentes-Pacheco, J., Ruiz-Ascencio, J., Rendón-Mancha, J.M.: Visual simultaneous localization and mapping: a survey. Artificial Intelligence Review 43(1), 55–81 (2015)
2. Meyer, J., Filliat, D.: Map-based navigation in mobile robots. II. A review of map-learning and path-planning strategies. Cognitive Systems Research 4(4), 283–317 (2003)
3. Milford, M., Wyeth, G., Prasser, D.: RatSLAM: a hippocampal model for simultaneous localization and mapping. In: Proceedings of the 2004 IEEE International Conference on Robotics and Automation, ICRA 2004, April 26 - May 1, 2004, New Orleans, LA, USA. pp. 403–408 (2004)
4. Milford, M., Wyeth, G.: Persistent navigation and mapping using a biologically inspired SLAM system. I. J. Robotics Res. 29(9), 1131–1153 (2010)
5. Wyss, R., König, P., Verschure, P.F.M.J.: A model of the ventral visual system based on temporal stability and local memory. PLoS Biol 4, e120 (04 2006)
6. Franzius, M., Sprekeler, H., Wiskott, L.: Slowness and sparseness lead to place, head-direction, and spatial-view cells. PLoS Computational Biology 3(8), 1–18 (08 2007)
7. Wiskott, L., Sejnowski, T.: Slow feature analysis: Unsupervised learning of invariances. Neural Computation 14(4), 715–770 (2002)
8. Metka, B., Franzius, M., Bauer-Wersing, U.: Outdoor self-localization of a mobile robot using slow feature analysis. In: Neural Information Processing - 20th International Conference, ICONIP 2013, Daegu, Korea, November 3-7, 2013. Proceedings, Part I. pp. 249–256 (2013)

9. Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science and Cybernetics SSC-4(2), 100–107 (1968)
10. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. In: Proceedings. 1985 IEEE International Conference on Robotics and Automation. vol. 2, pp. 500–505 (Mar 1985)
11. Barraquand, J., Latombe, J.: Robot motion planning: A distributed representation approach. I. J. Robotics Res. 10(6), 628–649 (1991)
12. Tilove, R.B.: Local obstacle avoidance for mobile robots based on the method of artificial potentials. In: Proceedings., IEEE International Conference on Robotics and Automation. pp. 566–571 vol.1 (May 1990)
13. Legenstein, R., Wilbert, N., Wiskott, L.: Reinforcement learning on slow features of high-dimensional input streams. PLoS Computational Biology 6(8), 1–13 (2010)
14. Böhmer, W., Grünewälder, S., Shen, Y., Musial, M., Obermayer, K.: Construction of approximation spaces for reinforcement learning. Journal of Machine Learning Research 14(1), 2067–2118 (2013)
15. Metka, B., Franzius, M., Bauer-Wersing, U.: Efficient navigation using slow feature gradients. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017. pp. 1311–1316 (2017)
16. Zito, T., Wilbert, N., Wiskott, L., Berkes, P.: Modular toolkit for data processing (mdp): a python data processing framework. Frontiers in Neuroinformatics 2(8) (2009)
17. Fox, D., Burgard, W., Dellaert, F., Thrun, S.: Monte carlo localization: Efficient position estimation for mobile robots. In: Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence. pp. 343–349 (1999)
18. Ishiguro, H., Tsuji, S.: Image-based memory of environment. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS 1996, November 4-8, 1996, Osaka, Japan. pp. 634–639 (1996)
19. Richthofer, S., Wiskott, L.: Global navigation using predictable and slow feature analysis in multiroom environments, path planning and other control tasks. CoRR (2018)
20. Metka, B., Bauer-Wersing, U., Franzius, M.: Improving robustness of slow feature analysis based localization using loop closure events. In: International Conference on Artificial Neural Networks (ICANN). Springer (2016)
21. Haris, M., Metka, B., Franzius, M., Bauer-Wersing, U.: Condition invariant visual localization using slow feature analysis. In: New Challenges in Neural Computation (NC2). Machine Learning Reports, Frank-Michael Schleif (September 2017)