Unsupervised Learning of Metric Representations with Slow Features from Omnidirectional Views

Mathias Franzius
Honda Research Institute Europe GmbH
63073 Offenbach, Germany
Mathias.Franzius@honda-ri.de

Benjamin Metka
Frankfurt University of Applied Sciences
60318 Frankfurt, Germany
benjamin.metka@gmobis.com

Muhammad Haris
Frankfurt University of Applied Sciences
60318 Frankfurt, Germany
muhammad.haris@fb2.fra-uas.de

Ute Bauer-Wersing
Frankfurt University of Applied Sciences
60318 Frankfurt, Germany
ubauer@fb2.fra-uas.de

Abstract

Unsupervised learning of Self-Localization with Slow Feature Analysis (SFA) using omnidirectional camera input has been shown to be a viable alternative to established SLAM approaches. Previous models for SFA self-localization purely relied on omnidirectional visual input. The model led to globally consistent localization in SFA space but the lack of odometry integration reduced the local accuracy. However, odometry integration and other downstream usage of localization require a common coordinate system, which previously was based on an external metric ground truth measurement system. Here, we show an autonomous unsupervised approach to generate accurate metric representations from SFA outputs without external sensors. We assume locally linear trajectories of a robot, which is consistent with, for example, driving patterns of robotic lawn mowers. This geometric constraint allows a formulation of an optimization problem for the regression from slow feature values to the robot's position. We show that the resulting accuracy on test data is comparable to supervised regression based on external sensors. Based on this result, using a Kalman filter for fusion of SFA localization and odometry is shown to further increase localization accuracy over the supervised regression model.

1. Introduction

A model of hierarchical Slow Feature Analysis (SFA) enables a mobile robot to learn a spatial representation of its environment directly from omnidirectional images taken by the robot (for details see [1]). In previous work it has been shown that bio-inspired SFA-based localization can be successfully and robustly applied to real world scenarios in indoor and outdoor environments showing comparable accuracy to state-of-the-art monocular visual SLAM approaches [1]. Moreover, extension of the basic model [2] led to improved robustness of localization in respect to medium- and long-term changes in outdoor scenes [3, 4]. Gradient-based navigation in slow feature space has proven feasible even under the constraint of obstacle avoidance [5, 6].

After an unsupervised learning phase using simulated rotation based on omnidirectional views the resulting SFA representations are orientation invariant and code for the position of the robot. The slowest features are spatially smooth (and in the theoretical optimum monotonous [7]) even though the variation of the sensory signals received from the environment might change drastically e.g., during rotation on the spot. Our model of spatial localization learning, like other models for localization based on slowness learning (e.g., [5, 8, 9]), does not integrate self-motion cues over time. It instantaneously estimates the absolute position in slow feature space from a single image. Hence, our model is complementary to path integration, which is locally consistent but drifts over time.

An appropriate combination of relative internal and absolute external measurements can improve localization accuracy compared to estimations based on the individual ones alone. This is a common approach in SLAM methods where the trajectory is estimated incrementally based on ego-motion estimates and accumulated errors are corrected using information from loop closure detections, i.e., the robot identifies a place it has seen before [10].

The combination of slow features with self-motion cues requires a common coordinate system. Earlier publications (e.g., [3, 5]) used supervised regression to map from slow

features to metric coordinates. The necessary ground truth position information was obtained from visual pose estimations of fiducial markers attached to the robot with external static cameras. However, the use of additional external infrastructure is costly, complex, and often causes synchronization issues. Furthermore in many robotic application scenarios e.g. household or mower robots it is impractical to obtain external ground truth. Therefore, we introduce a method to learn the mapping function from slow feature space to metric space in an unsupervised fashion.
Section 2 briefly summarizes Slow Feature Analysis (SFA) and shows how hierarchical spatial representation is learned from omnidirectional image data. In Section 3 we introduce unsupervised metric learning and then show experimental results on images from a simulator and from a mobile robot in Section 4. Section 5 extends the model by fusing the metric localization data with odometry by Kalman filtering and shows experimental data on improved localization accuracy.

2. Learning Spatial Representations with SFA

2.1. Slow Feature Analysis

Slow feature analysis (SFA) [11] transforms a multidimensional time series $\mathbf{x}(t)$, in our case images along a trajectory, to slowly varying output signals. The objective is to find instantaneous scalar input-output functions $g_j(\mathbf{x})$ such that the output signals

$$s_j(t) := g_j(\mathbf{x}(t))$$

minimize

$$\Delta(s_j) := \langle \dot{s}_j^2 \rangle_t$$

under the constraints

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

with $\langle \cdot \rangle_t$ and $\dot{s}$ indicating temporal averaging and the derivative of s, respectively. The Δ-value is a measure of the temporal slowness of the signal $s_j(t)$. It is given by the mean square of the signal's temporal derivative, so small Δ-values indicate slowly varying signals. The constraints avoid the trivial constant solution and ensure that different functions g_j code for different aspects of the input. We use the MDP [12] implementation of SFA, which is based on solving a generalized eigenvalue problem.

2.2. Learning Orientation Invariant Spatial Representations

For representations that are suitable for self-localization we want to learn functions that encode the robot‘s position as slowly varying features and are invariant w.r.t. its orientation. Since the information encoded in the learned slow features depends on the temporal statistics of the training data, the orientation of the robot has to change on a faster timescale than its position during training in order to achieve orientation invariance [7]. To avoid a trajectory with excessive robot rotation, we simulate additional robot rotation by virtually rotating the omnidirectional images from the robot's camera [2].
The high dimensional visual input is processed by a hierarchical converging network. In our standard approach [2], layers are trained subsequently with all training images and their variants from the simulated rotation, in temporal order of their recording. The layers converge onto a single node whose slowest outputs $s_{1...K}$ form the representation of the environment that we use for localization (see Fig. 1). Please note that during the training phase simulated rotation is essential for learning orientation invariant position. After training, localization itself is instantaneous i.e. position is predicted based on a single omnidirectional view.
This first phase of the algorithm for learning the mapping of omnidirectional views to slow feature space is also depicted as the left part of the overall processing pipeline shown in Fig. 2.

3. Unsupervised Metric Learning

Given the slow feature vector $\mathbf{s} \in \mathbb{R}^K$ computed for the image from a given position $\mathbf{p} := (x, y)^\top$ we want to find the weight matrix $\mathbf{W} = (\mathbf{w^x}\ \mathbf{w^y}) \in \mathbb{R}^{K \times 2}$ such that the error $\varepsilon \in \mathbb{R}^2$ for the estimation $\mathbf{p} = \mathbf{W}^\top \mathbf{s} + \varepsilon$ is minimal. Without external measurements of the robot's position $\mathbf{p}$ the only source of metric information available is from ego-motion estimates. As already stated, pose estimation solely based on odometry accumulates errors over time and thus does not provide suitable label information to learn the weight matrix $\mathbf{W}$ directly. Especially errors in the orientation measurements cause large deviations. The odometry distance measurements, on the other hand, are locally very precise. In order to learn the weight matrix $\mathbf{W}$ using these distance measurements, we assume particular trajectory forms driven by the robot. Here, we require the robot trajectory to contain straight line segments, such that the training area is evenly covered and there exists a certain number of intersections between the lines. We consider this a minimal constraint since such movement strategy is typical for current household robots. This second phase phase for learning the mapping from slow feature space to 2D coordinates in metric space is depicted in right part of the over-

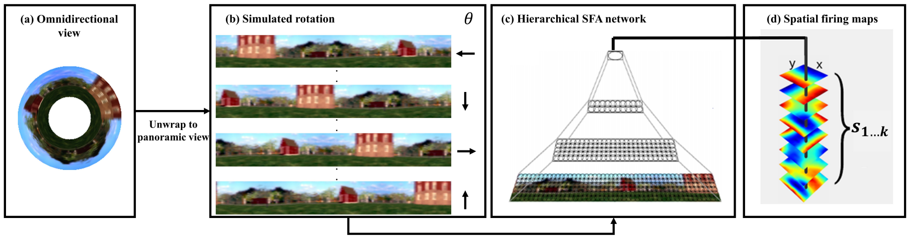

Figure 1: **Learning the spatial representation**. (a) An omnidirectional view from the current position on the training trajectory is captured and transformed to a panoramic view. (b) A full rotation of the robot is simulated for every image by circular shifts of the panoramic view, which increases the perceived rotational movement and leads to orientation invariant encoding of the position. (c) The views are processed by the network. Each layer consists of overlapping SFA-nodes arranged on a regular grid. Each node performs linear SFA for dimensionality reduction followed by a quadratic SFA for slow feature extraction. The output layer is a single SFA-node. (d) The color-coded outputs, so-called spatial firing maps, ideally show characteristic gradients along the coordinate axes and look the same independent of the specific orientation. Thus, SFA-outputs $s_{1...K}$ at position *p* are the orientation invariant encoding of location.

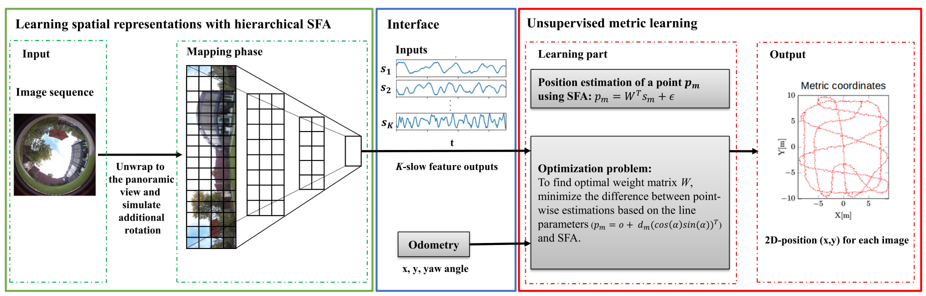

Figure 2: **SFA and Unsupervised Metric Learning**. The input to the processing pipeline are omnidirectional views of the environment and the *first phase* starts with an initial step that projects all the images to panoramic views and simulates additional rotation to ensure orientation invariant slow features. The hierarchical SFA network then uses these images to learn spatial representations of the environment in SFA space/coordinates. While traversing the training trajectory the m SFA outputs change over time hence encode the robot's changing position in SFA space, indicated as *interface*. In the *second phase* the unsupervised metric learning (UML) module uses these outputs of the trained SFA network together with odometry information as inputs to learn the metric coordinates. The UML module determines a mapping from SFA to metric space by minimizing the point-wise estimation based on the line parameters and SFA outputs. The overall output of the system is 2D-position (x,y) for each omnidirectional image.

all processing pipeline shown in Fig.2.

3.1. Definition of the Cost Function

A line l consists of M collinear points $\mathbf{P} = (\mathbf{p}_1, \dots, \mathbf{p}_M)$. At every point $\mathbf{p}_m$ we record the slow feature vector $\mathbf{s}_m$ computed for the corresponding image and the current distance measurement d_m to the origin $\mathbf{o}$ of line l where $d_m = ||\mathbf{p}_m - \mathbf{o}||_2$ and $\mathbf{o} := (x_0, y_0)^\top$. Based on the orientation α, the origin $\mathbf{o}$ and the distance measurement d_m the reconstruction of a point is given by the equation $\mathbf{p}_m = \mathbf{o} + d_m(\cos(\alpha)\ \sin(\alpha))^\top$. For the correct weight matrix $\mathbf{W}$ the reconstruction of the same point using slow feature vector $\mathbf{s}_m$ is defined by $\mathbf{p}_m = \mathbf{W}^\top \mathbf{s}_m$. However, the line parameters $\mathbf{o}$ and α as well as the weight matrix $\mathbf{W}$ are unknown and need to be estimated. Given optimal parameters the difference between the point-wise estimations based on the line parameters and the weight matrix are minimal. Thus, the parameters can be learned simultaneously by minimizing the difference in the point-wise reconstruction (see Fig. 3).

The distance measurements from odometry induce the correct metric scale while the intersections of the line segments and the weights ensure a globally consistent mapping. For N line segments the cost function for parameters $\theta = (\alpha_1, ..., \alpha_N, \mathbf{o}_1, ..., \mathbf{o}_N, \mathbf{W})$ is the following:

$$C(\theta) = \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{M_n} ||\mathbf{o}_n + d_{n,m} \Big(cos(\alpha_n) \quad sin(\alpha_n) \Big)^\top - \mathbf{W}^\top \mathbf{s}_{n,m}||_2^2 \tag{1}$$

Where M_n is the number of points on line l_n. The corresponding partial derivatives w.r.t. parameters θ are given

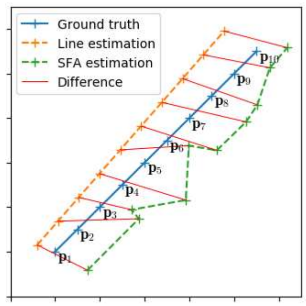

Figure 3: **Optimization problem.** Find the optimal values for weight matrix $\mathbf{W}$ by minimizing the difference between estimations based on the line parameters and SFA.

by:

$$\frac{\partial C}{\partial \alpha_n} = \sum_{m=1}^{M} d_{n,m} \begin{pmatrix} sin(\alpha_n) & -cos(\alpha_n) \end{pmatrix} (\mathbf{W}^\top \mathbf{s}_{n,m} - \mathbf{o}_n) \tag{2}$$

$$\frac{\partial C}{\partial \mathbf{o}_n} = \sum_{m=1}^{M} \mathbf{o}_n + d_{n,m} \begin{pmatrix} cos(\alpha_n) & sin(\alpha_n) \end{pmatrix}^\top - \mathbf{W}^\top \mathbf{s}_{n,m} \tag{3}$$

$$\frac{\partial C}{\partial \mathbf{W}} = \sum_{n=1}^{N} \sum_{m=1}^{M} -\mathbf{s}_{n,m} (\mathbf{o}_n + d_{n,m} \begin{pmatrix} cos(\alpha_n) & sin(\alpha_n) \end{pmatrix}^\top - \mathbf{W}^\top \mathbf{s}_{n,m})^\top \tag{4}$$

3.2. Extension of the Cost Function

For some trajectories, e.g., a grid-like trajectory, the learned mapping can be sheared relative to the ground truth. In the most extreme case, the learned solution maps all points onto a single line. To avoid this problem, the orientations α_n and α_{n+1} of two consecutive line segments l_n, l_{n+1} can be constrained such that the relative angle between them corresponds to the measured change in orientation from odometry. Therefore, we add a term to the cost function C that punishes the deviation of the relative angle defined by the current estimate of α_n and α_{n+1} from the measured change in orientation obtained from odometry $\measuredangle l_n l_{n+1}$. We express α_n and α_{n+1} as unit vectors to obtain the cosine of the relative angle given by their dot product. The deviation of the relative angle from the measured angle is then defined as the difference between the angles' cosine values. The cost function C from Eq. 1 is thus extended:

$$\begin{aligned} C'(\theta) = \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{M} ||\mathbf{o}_n + d_{n,m} \begin{pmatrix} cos(\alpha_n) & sin(\alpha_n) \end{pmatrix}^\top - \mathbf{W}^\top \mathbf{s}_{n,m}||_2^2 \\ + \frac{1}{2} \sum_{n=1}^{N-1} (\cos(\alpha_n)\cos(\alpha_{n+1}) + \sin(\alpha_n)\sin(\alpha_{n+1}) - \cos(\measuredangle l_n l_{n+1}))^2 \end{aligned} \tag{5}$$

Thus, while C only uses translation measurements, C' additionally uses yaw rotation information from odometry. The partial derivatives of cost function C' are equal to those of the cost function C except for the partial derivative of α_n:

$$\begin{aligned} \frac{\partial C'}{\partial \alpha_n} &= \sum_{m=1}^{M} d_{n,m} \begin{pmatrix} sin(\alpha_n) & -cos(\alpha_n) \end{pmatrix} (\mathbf{W}^\top \mathbf{s}_{n,m} - \mathbf{o}_n) \\ &+ \begin{cases} \sin(\alpha_n - \alpha_{n+1})(\cos(\measuredangle l_n l_{n+1}) - \cos(\alpha_n - \alpha_{n+1}), & \text{if } n = 1 \\ \sin(\alpha_n - \alpha_{n+1})(\cos(\measuredangle l_n l_{n+1}) - \cos(\alpha_n - \alpha_{n+1}) & \\ + \sin(\alpha_{n-1} - \alpha_n)(\cos(\alpha_{n-1} - \alpha_n) - \cos(\measuredangle l_{n-1} l_n), & \text{otherwise} \end{cases} \end{aligned} \tag{6}$$

A solution for the parameters θ can be obtained performing gradient descent on the cost function C. The update in an iteration step t is given by:

$$v_t = \gamma v_{t-1} + \beta \frac{\partial C}{\partial \theta} \tag{7}$$

$$\theta_t = \theta_{t-1} - v_t \tag{8}$$

Where β is the learning rate and $\gamma \in (0, 1]$ is a momentum term to increase speed of convergence and regulate the amount of information from previous gradients which is incorporated into the current update. The right part of Fig. 2 shows the pipeline of unsupervised metric learning.

Note that the found solutions may be translated and rotated against the odometry's coordinate systems and for cost function C it may also be mirrored. If required, some parameters $\mathbf{o}$ and α can be fixed during the optimization to be compatible with the desired coordinate system or by rotating, shifting, and mirroring the solution as a postprocessing step.

4. Experiments

The overall processing pipeline is summarized in Fig. 2. We assume that the slow feature representation is learned in advance from omnidirectional views of the environment. Subsequently slow feature outputs and odometry recordings are used to estimate the mapping into metric space. The optimal parameters for unsupervised metric learning are obtained by minimizing the cost functions given in equations 1 and 5 using the distances d and slow feature vectors $\mathbf{s}$ sampled along the line segments. Optimization terminates if either the number of maximum iterations is reached or the change in the value of the cost function falls below a threshold. To assess the quality of the learned metric mapping the localization accuracy was measured on a separate test set as the mean Euclidean deviation from the ground truth coordinates. As a reference, the results of training a regression model directly on the ground truth coordinates have been computed as well. Since there is no point of reference between both coordinate systems the estimated coordinates might be rotated and translated. Therefore, the ground truth and estimated metric coordinates have to be aligned before calculating the accuracy. We used the method from [13] to obtain the rigid transformation which rotates and translates the estimated coordinates to align them with the ground truth coordinates. The obtained transformation was then again applied to the estimations from our separate test set. We used the eight slowest features for regression. A non-linear expansion of the slow features to all monomials of degree 2 yields a 45-dimensional representation, which slightly increases localization accuracy. The number of unknown parameters per line is $1 + 2$ (scalar α, 2D line offset $\mathbf{o}$). Additional, the regression weights $\mathbf{W}$ for two dimensions x, y with $2 * 45$ dimensions are unknown parameters.

Simulator Experiment The approach was first validated in a simulated garden-like environment created with Blender as in [5]. The spatial representation was learned by training the SFA model with 1773 panoramic RGB-images with a resolution of 600×60 pixels from a trajectory that covers an area of 16×18 meters. Training data for the unsupervised metric learning was gathered by driving along 10 straight line segments with a random orientation and sampling the slow feature vector $\mathbf{s_{n,m}}$, the distance measurement $d_{n,m}$ and the corresponding ground truth coordinates $(x_{n,m}, y_{n,m})^\top$ in 0.2m steps. In total 862 points were collected. Start and end points of the line segment were restricted to be within the training area and to have a minimum distance of 8m. The parameters for the optimization were initialized with values from a random uniform distribution such that $\alpha_n \in [0, 2\pi)$, $\mathbf{o}_n^x \in [-8, 8]$, $\mathbf{o}_n^y \in [-9, 9]$ and $\mathbf{w}_j^x, \mathbf{w}_j^y \in [-1, 1]$. The learning rate was set to $\beta = 1 \times 10^{-5}$ and the momentum term to $\gamma = 0.95$. The partial derivatives of the cost function are computed according to equations (2)-(4).

Results The optimization terminated at about 1000 iterations where the change in the value of the cost function fell below the predefined threshold. The process starting from random parameters over intermediate results to convergence is illustrated in Fig. 4.

As a baseline for the training line segments, the mean Euclidean deviation from ground truth to the estimated coordinates from the supervised regression model amounts to 0.13m. The unsupervised metric learning on the training data results in an error of 0.17m. Both estimations for the line segments from the training set are illustrated in Fig. 5.

Using the weights learned with the supervised regression model to predict coordinates on the separate test trajectory results in a mean Euclidean deviation of 0.39m from ground truth. The unsupervised model prediction error is slightly lower with a mean deviation of 0.36m. Predicted trajectories from both models closely follow the ground truth coordinates with noticeable deviations in the south-eastern and north-western part. Considering the line segments from the training data it is apparent that those regions are only sparsely sampled while the density is much higher in the middle-eastern part. The lower accuracy of the supervised regression model might be due to slightly overfitting to the training data. The estimated trajectories of the supervised and unsupervised regression models are shown in Fig. 6.

Robot We recorded omnidirectional image data with camera mounted on an autonomous lawn mower together with the wheel odometry measurements from the robot and the yaw angle estimated by a gyroscope. To obtain the training data for the unsupervised metric learning we used the same model architecture as in the previous section and trained it while driving along straight lines, turning on the spot and driving along the next straight line. Thus, the points on the trajectory could easily be split into line segments based on the translational and angular velocity measurements from odometry. There were 18 line segments consisting of a total of 1346 points. In order to speed up the process and support convergence to an optimal solution, the line parameters α_n and $\mathbf{o}_n$ have been initialized with the corresponding odometry measurements. The weight vectors $\mathbf{w}^x$ and $\mathbf{w}^y$ were initialized with the weights of regression models fitted to the raw odometry estimations. As in the simulator experiment the learning rate is set to $\beta = 1 \times 10^{-5}$ and the momentum term to $\gamma = 0.95$. Due to the grid-like trajectory with intersection angles all close to $90°$ the cost function C' from equation (5) was used for the optimization.

Results The optimization ran for about 900 iterations until it converged to a stable solution. The localization accuracy of the unsupervised model on the training data amounts to 0.17m after estimations have been aligned to the ground truth coordinates. The supervised model which was trained directly with the ground truth coordinates achieved an accuracy of 0.12m. An illustration of the estimations for the training data from both models is shown in Fig. 7. Estimations from both models for the test data have the same mean Euclidean deviation of 0.14m. The resulting predicted trajectories along with the ground truth trajectories are shown in Fig. 8.

5. Fusion of SFA Estimates and Odometry in a Probabilistic Filter

In our scenario the robot has access to relative motion measurements from odometry and absolute measurements from the SFA-model in order to localize itself. Even though the odometry measurements are locally very precise small errors accumulate over time and the belief of the own position starts to diverge from the true position. The estimations from the SFA-model on the other hand have a higher variability but are absolute measurements and thus allow to correct for occurring drift. The mapping from slow feature to metric space from the previous section enables the combi-

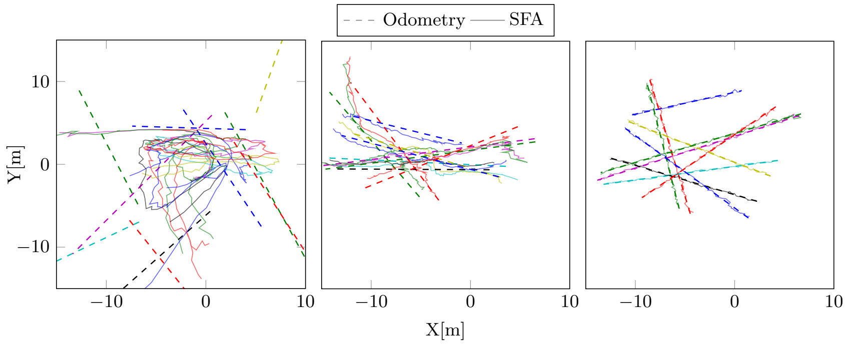

Figure 4: **Optimization process on simulator training data** (a) Metric position estimates with randomly initialized parameters α_n, $\mathbf{o}_n$, $\mathbf{W}$. Each color visualizes a line segment (dashed: ground truth position, solid: metric estimate). (b) Estimates after 500 iterations. (c) At iteration 1000 the estimates converged.

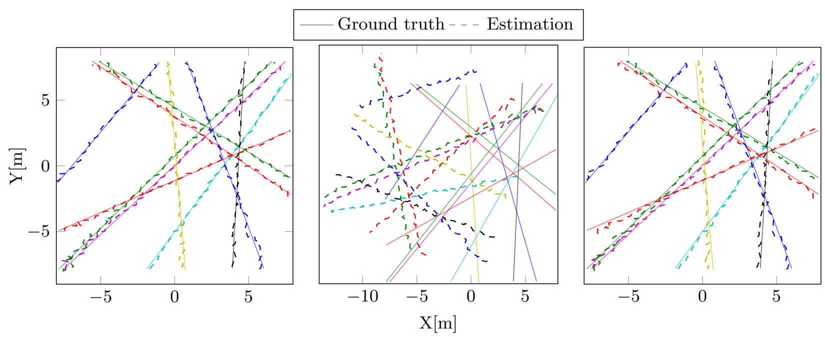

Figure 5: **Comparison of supervised and unsupervised position estimation on simulator training data.** (a) Estimated positions from supervised regression are close to the ground truth with a mean Euclidean distance of 0.13m. (b) Estimations resulting from the unsupervised learned regression weights are consistent but rotated and translated with respect to the ground truth coordinates. (c) After aligning the estimated coordinates to the ground truth the mean Euclidean distance amounts to 0.17m.

nation of both measurements in a common coordinate system. For linear systems with uncorrelated white noise the Kalman Filter [14] is the optimal estimator, using a state transition model to predict the value for the next time step and a measurement model to correct the value based on observations. The state of the system is represented as a Gaussian distribution. However, for our scenario of mobile robot localization the state transition model involves trigonometric functions which lead to a nonlinear system. The *Extended Kalman Filter* (EKF) linearizes the state transition and measurement model around the current estimate to ensure the distributions remain Gaussian. Then, the equations of the linear Kalman Filter can be applied. This is the standard method for the problem of vehicle state estimation [15] and is also applied in Visual SLAM [16]. Here, we use the Constant Turn Rate Velocity (CTRV) model [17] as the state transition model for the mobile robot. The measurement model incorporates the absolute estimations resulting from the mapping of the current slow feature outputs to metric coordinates.

5.1. Robot Experiment

To test the localization performance with the EKF we used the test data from the robot experiment in the previous section. The absolute coordinate predictions from slow feature outputs were computed using the unsupervised learned regression weights from the corresponding training data. They are used as input for the measurement model while the odometry readings are used as input for the state transition model of the EKF. The values for the process and measurement noise covariance matrices were chosen based on a grid search. For the experiment we assumed that the

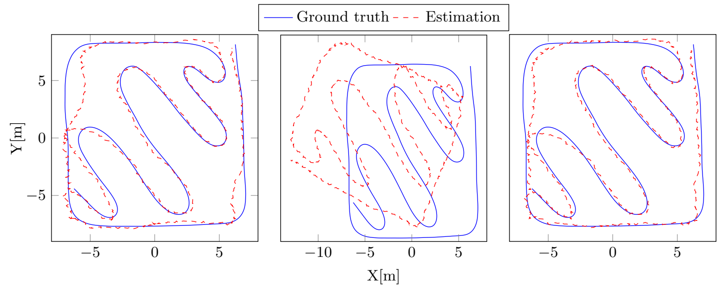

Figure 6: **Comparison of supervised and unsupervised regression for simulator test data.** (a) Estimated trajectory for test data using supervised regression results in an average error of 0.39m. (b) The estimations of the unsupervised regression model are not aligned with the coordinate system of the ground truth coordinates. (c) Applying the aligning transformation obtained for the training data to the test set results in an error of 0.36m. Both estimates closely follow the true trajectory while the supervised model may have slightly overfitted on the training data.

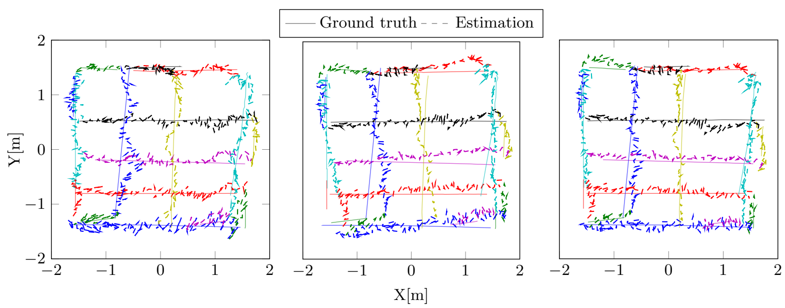

Figure 7: **Comparison of supervised and unsupervised regression for robot training data.** Each straight line segment is illustrated in a separate color where the solid lines represent the ground truth and dotted lines indicate the estimations of the respective models. (a) The predictions of the supervised model have an error of 0.12m closely following the ground truth. (b) The raw predictions of the unsupervised model are slightly rotated and shifted w.r.t. the ground truth coordinates. (c) Aligning the estimations from the unsupervised model to the ground truth coordinates results in a mean Euclidean deviation of 0.17m.

robot starts from a known location and with known heading which is a valid assumption considering that many service robots begin operating from a base station.

Results As expected, the estimated trajectory of the EKF shows an improvement over the individual estimations since it combines their advantages of global consistency and local smoothness. The accuracy of the predicted trajectory from the SFA-model is 0.14m with noticeable jitter. The trajectory resulting from odometry measurements is locally smooth but especially the errors in the orientation estimation lead to a large divergence over time resulting in a mean Euclidean deviation of 0.31m from the ground truth coordinates. The accuracy of the trajectory estimated from the EKF amounts to 0.11m which is an improvement of 21% on average compared to the accuracy obtained from the SFA-model and an improvement of 65% compared to odometry. Resulting trajectories from all methods are illustrated in Fig. 9.

6. Discussion

The presented method for unsupervised learning of a mapping from slow feature outputs to metric coordinates was successfully applied in simulator and real world robot experiments and achieved accuracies similar to supervised regression trained directly on ground truth coordinates. Since it only requires omnidirectional images and odometry measurements and imposes reasonable constraints on the trajectory, it can be applied in real application scenarios where no external ground truth information is available.

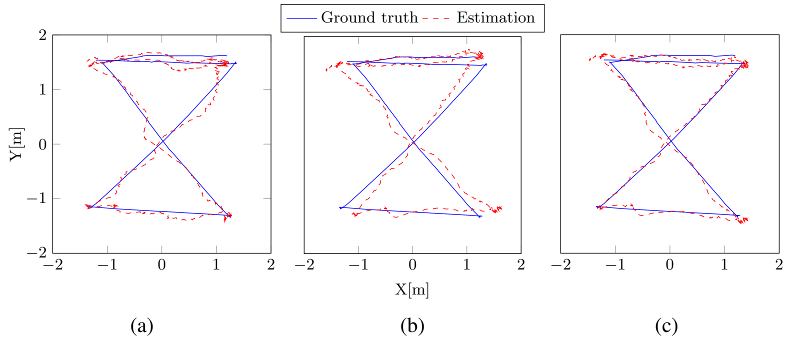

Figure 8: **Comparison of supervised and unsupervised regression for robot test data.** (a) The trajectory predicted from the supervised model deviates by 0.14m on average from the ground truth trajectory. (b) The raw estimations from the unsupervised model are rotated and translated w.r.t. the ground truth trajectory. (c) Transforming the estimations from the supervised model by the rotation and translation estimated for the training data results in a mean Euclidean deviation of 0.14m from the ground truth.

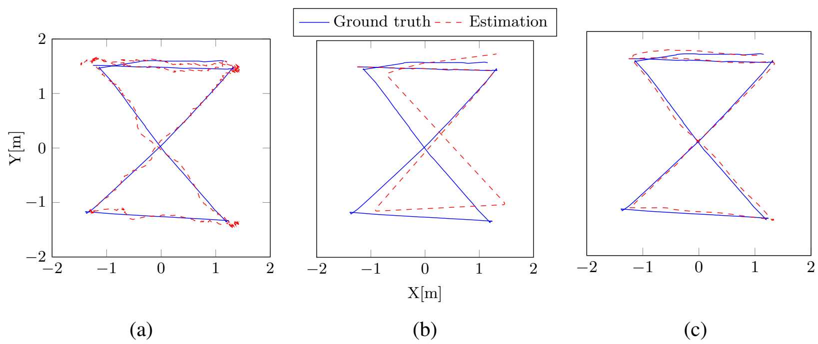

Figure 9: **Fusion of SFA estimates and odometry using an Extended Kalman Filter.** (a) The accuracy of the localization achieved with the SFA-model is 0.14m. Due to the absolute coordinate predictions the progression of the trajectory is rather erratic. (b) Measurements from odometry are locally accurate but increasingly diverge over time. The mean Euclidean deviation from ground truth amounts to 0.31m. (c) The EKF filter combines the strength of both estimations resulting in an accuracy of 0.11m.

The learned metric mapping function enables the visualization of the extracted slow feature representations, the trajectories of the mobile robot and the fusion of SFA estimates and odometry measurements using an Extended Kalman Filter. Thereby, the already competitive localization accuracy of the SFA-model trained on monocular omnidirectional images improved further by 21%. Please note that the relative improvement in accuracy will be even more pronounced for longer trajectories.

In contrast to geometric approaches, appearance based localization can cope with uncalibrated cameras. Other appearance based localization models like [18, 19] are usually supervised (requiring ground truth positions) or can not cope with arbitrary camera orientations, which we solved with the orientation invariant representations learned from omnidirectional views by SFA.

The proposed approach may also be useful for other spatial representations than those learned by SFA, as long as they are sufficiently low-dimensional, spatially smooth, and preferably orientation invariant. The precision of the resulting metric mapping, and hence also the localization accuracy, might be further improved using visual odometry [20–22] instead of wheel odometry since it is not affected by wheel slippage.

We have shown in [5] how navigation directly in slow feature space allows, for example, to navigate around known obstacles without explicit planning. Thus, while unsupervised mapping of SFA features to metric space is not strictly necessary, it improves transparency and allows better integration with other methods and services based on metric location.

References

[1] Benjamin Metka, Mathias Franzius, and Ute Bauer-Wersing. Bio-inspired visual self-localization in real world scenarios using slow feature analysis. *PLOS ONE*, 13(9):1–18, 09 2018. doi: 10.1371/journal.pone.0203994. URL `https://doi.org/10.1371/journal.pone.0203994`.

[2] Benjamin Metka, Mathias Franzius, and Ute Bauer-Wersing. Outdoor self-localization of a mobile robot using slow feature analysis. In *Neural Information Processing - 20th International Conference, ICONIP 2013. Proceedings, Part I*, volume 8226, pages 249–256. Springer, 2013. doi: 10.1007/978-3-642-42054-2_32. URL `https://doi.org/10.1007/978-3-642-42054-2_32`.

[3] Benjamin Metka, Mathias Franzius, and Ute Bauer-Wersing. Improving robustness of slow feature analysis based localization using loop closure events. In *ICANN 2016*, pages 489–496. Springer International Publishing, 2016.

[4] Muhammad Haris, Mathias Franzius, and Ute Bauer-Wersing. Robust outdoor self-localization in changing environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2019)*. IEEE, 2019.

[5] Benjamin Metka, Mathias Franzius, and Ute Bauer-Wersing. Efficient navigation using slow feature gradients. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017*, pages 1311–1316, 2017.

[6] Muhammad Haris, Mathias Franzius, and Ute Bauer-Wersing. Robot navigation on slow feature gradients. In *Neural Information Processing*, pages 143–154. Springer International Publishing, 2018. ISBN 978-3-030-04239-4.

[7] Mathias Franzius, Henning Sprekeler, and Laurenz Wiskott. Slowness and Sparseness Lead to Place, Head-Direction, and Spatial-View Cells. *PLoS Computational Biology*, 3(8):1–18, 2007.

[8] Reto Wyss, Peter König, and Paul F M J Verschure. A Model of the Ventral Visual System Based on Temporal Stability and Local Memory. *PLoS Biol*, 4:e120, 2006.

[9] Eric Antonelo and Benjamin Schrauwen. Learning slow features with reservoir computing for biologically-inspired robot localization. *Neural Networks*, 25:178 – 190, 2012. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2011.08.004.

[10] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: part i. *IEEE Robotics Automation Magazine*, 13(2):99–110, June 2006. ISSN 1558-223X. doi: 10.1109/MRA.2006.1638022.

[11] Laurenz Wiskott and Terrence Sejnowski. Slow Feature Analysis: Unsupervised Learning of Invariances. *Neural Computation*, 14(4):715–770, 2002.

[12] Tiziano Zito, Niko Wilbert, Laurenz Wiskott, and Pietro Berkes. Modular toolkit for Data Processing (MDP): a Python data processing framework. *Front. Neuroinform.*, 2 (8), 2009. ISSN 1662-5196. doi: 10.3389/neuro.11.008.2008.

[13] K S Arun, Thomas S Huang, and Steven D Blostein. Least-Squares Fitting of Two 3-D Point Sets. *IEEE Trans. Pattern Anal. Mach. Intell.*, 9(5):698–700, 1987.

[14] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME – Journal of Basic Engineering*, (82 (Series D)):35–45, 1960.

[15] Alonzo Kelly. A 3d state space formulation of a navigation Kalman filter for autonomous vehicles. 06 2000.

[16] Andrew J Davison, Ian D Reid, Nicholas Molton, and Olivier Stasse. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(6):1052–1067, 2007. doi: 10.1109/TPAMI.2007.1049.

[17] X. Rong Li and V. P. Jilkov. Survey of maneuvering target tracking. part I. dynamic models. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1333–1364, Oct 2003. ISSN 0018-9251. doi: 10.1109/TAES.2003.1261132.

[18] Jihun Ham, Yuanqing Lin, and D. D. Lee. Learning nonlinear appearance manifolds for robot localization. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2971–2976, Aug 2005.

[19] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 2938–2946, 2015. doi: 10.1109/ICCV.2015.336. URL `https://doi.org/10.1109/ICCV.2015.336`.

[20] David Nistér, Oleg Naroditsky, and James R. Bergen. Visual odometry. In *2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004), 27 June - 2 July 2004, Washington, DC, USA*, pages 652–659, 2004. doi: 10.1109/CVPR.2004.265.

[21] D. Scaramuzza and F. Fraundorfer. Visual Odometry : Part I: The First 30 Years and Fundamentals. *IEEE Robotics Automation Magazine*, 18(4):80–92, Dec 2011. ISSN 1070-9932. doi: 10.1109/MRA.2011.943233.

[22] F Fraundorfer and D Scaramuzza. Visual Odometry : Part II: Matching, Robustness, Optimization, and Applications. *Robotics Automation Magazine, IEEE*, 19(2):78–90, jun 2012. ISSN 1070-9932. doi: 10.1109/MRA.2012.2182810.