Supplementary Material

Hyperdimensional computing as a framework for systematic aggregation of image descriptors

Peer Neubert and Stefan Schubert
Chemnitz University of Technology
firstname.lastname@etit.tu-chemnitz.de

Contents

1. Versatility of the proposed HDC framework **1**
1.1. Encoding image sequences 1
1.2. Encoding local image feature position and scale 2
1.3. How to switch to a different HDC architecture? 4

2. Detailed experimental results that were summarized in the paper **4**
2.1. Unordered aggregation of holistic descriptors 4
2.2. Detailed results for parameters n_x and n_y 5

3. Dataset details **6**

1. Versatility of the proposed HDC framework

Sec. 3.2 of the paper presents two examples how the HDC framework can be used to encode information from image features and claims that the same approach can also be used to encode other information. To support this claim, the following sections 1.1 and 1.2 provide details and evaluations how such encodings can be done for sequences of images or image feature positions together with feature scales. Further, Sec. 1.3 will list the required steps to switch to a different underlying HDC architecture.

1.1. Encoding image sequences

In mobile robot place recognition, database and/or query image sets are often arranged in temporal sequences. In particular, this holds for all dataset in our evaluation (cf. Sec. 3). Therefore, when comparing a query image to a database image, the similarity of previous (and if available also subsequent) query images to corresponding database images can provide additional information. A simple and effective approach to exploit this information is the core of the SeqSLAM [10] algorithm: Given a matrix S of pairwise similarities of a database and a query sequence, it averages similarities over short linear trajectories of length s in S. Basically, this is a (sparse) convolution of S.

The paper [11] presented a hyperdimensional computing approach to approximate this core algorithmic step of SeqSLAM. It can be considered a special case of the presented HDC framework from this paper. Instead of post-processing the $(m \times n)$ similarity matrix S, the HDC approach pre-processes the n database and m query descriptors individually in order to encode information about neighbored images in the sequence in the descriptor of each individual image. Depending on the relation of the framerate in the sequence and the relocalization rate, this can significantly reduce the number of required computations (if the framerate is much higher than the relocalization rate the reduction is by a factor of the order of the sequence length s).

Basically, this approximation of SeqSLAM can be computed by a version of equation (4) from the paper (that was used to encode local features with their poses), where the local features are replaced by holistic descriptors X_i and the feature position encodings by random vectors P_k that encode the position k in the encoded trajectory sequence. The results is a new

descriptor Y_i that accumulates all input descriptors X_{i+k} from a sequence of length $s = 2 \cdot d + 1$:

$$Y_i = \bigoplus_{k=-d}^{d} (X_{i+k} \otimes P_k) \tag{1}$$

The original SeqSLAM [10] algorithm includes additional computing steps, e.g. it evaluates a set of possible (still constant) velocities. Although such additional steps can also be implemented in the HDC framework, Table 1 shows that there is already a large benefit by the simple implementation of eq. 1 in the HDC-Seq approach. We report results for all standard holistic descriptors from the paper as well as the combination with the HDC-DELF encoding of local DELF features. Although there are a few sequence comparisons with a decreased performance (presumably, the constant velocity assumption of SeqSLAM is violated), the average performance improvement by this HDC implementation of the SeqSLAM core is considerable.

We want to emphasize the fact that the improvement is due to the additional sequential information, not due to the HDC implementation. The HDC implementation is an approximation of the original SeqSLAM but provides a potentially more efficient implementation (similar to how HDC-DELF approximates an exhaustive comparison of local features). On average, it provides slightly worse results than the costly SeqSLAM matrix post-processing. The approximation quality depends on the used feature. While for AN the mAP is almost identical, in worst case (which was NV) the mAP was 0.05 lower than for the matrix post-processing. The benefit of HDC-Seq is the potentially significant runtime improvement. However, as said before, there is only a considerable reduction in number of CPU instructions, if the framerate is higher than the relocalization rate (or if the dataset size is considerably larger than the number of dimensions in the HDC vector space $\mathbb{V}$).

Table 1: Evaluation of the HDC sequence encoding HDC-Seq. The table reports average precision. Arrows compare to the single-image version in the column to the left (they indicate similar performance or more than ± 5 % or ± 25 % performance change).

Dataset	**DB**	**Query**	**AN**		**NV**		**DV**		**DELG**		**HDC-DELF**	
			single	**HDC-Seq**	**single**	**HDC-Seq**	**single**	**HDC-Seq**	**single**	**HDC-Seq**	**single**	**HDC-Seq**
GardensPointWalking	day_right	day_left	0.46	0.80 ↑	0.99	**1.00** →	0.98	**1.00** →	0.95	**1.00** →	0.82	**1.00** ↗
GardensPointWalking	day_right	night_right	0.62	0.91 ↑	0.59	0.77 ↑	0.52	0.67 ↑	0.44	0.55 ↑	0.79	**0.97** ↗
GardensPointWalking	day_left	night_right	0.12	0.27 ↑	0.48	0.65 ↑	0.22	0.24 ↗	0.32	0.46 ↑	0.47	**0.83** ↑
OxfordRobotCar	2014-12-09-13-21-02	2015-05-19-14-06-38	0.77	0.74 →	0.89	0.88 →	0.85	0.82 →	0.70	0.68 →	**0.91**	0.89 →
OxfordRobotCar	2014-12-09-13-21-02	2015-08-28-09-50-22	0.41	0.50 ↗	0.66	0.68 →	0.62	0.43 ↓	0.23	0.26 ↗	**0.71**	0.67 ↘
OxfordRobotCar	2014-12-09-13-21-02	2014-11-25-09-18-32	0.67	0.49 ↓	**0.91**	0.79 ↘	0.90	0.79 ↘	0.68	0.55 ↘	0.82	0.72 ↘
OxfordRobotCar	2014-12-09-13-21-02	2014-12-16-18-44-24	0.27	0.61 ↑	0.11	0.33 ↑	0.11	0.05 ↓	0.12	0.12 ↗	**0.80**	0.79 →
OxfordRobotCar	2015-05-19-14-06-38	2015-02-03-08-45-10	0.84	0.89 ↗	**0.93**	0.88 ↘	0.33	0.69 ↑	0.72	0.69 →	0.78	0.86 ↗
OxfordRobotCar	2015-08-28-09-50-22	2014-11-25-09-18-32	0.34	0.42 ↗	0.59	0.55 ↘	0.46	0.43 ↘	0.38	0.38 →	**0.71**	0.70 →
SFUMountain	dry	dusk	0.54	0.76 ↑	0.48	0.93 ↑	0.79	0.94 ↗	0.34	0.61 ↑	0.81	**0.98** ↗
SFUMountain	dry	jan	0.40	0.72 ↑	0.22	0.58 ↑	0.63	0.90 ↑	0.10	0.28 ↑	0.57	**0.95** ↑
SFUMountain	dry	wet	0.42	0.80 ↑	0.40	0.92 ↑	0.75	0.94 ↑	0.25	0.55 ↑	0.75	**0.98** ↑
CMU	20110421	20100901	0.52	0.64 ↗	0.71	0.79 ↗	0.69	0.70 →	0.80	**0.84** →	0.75	0.80 ↗
CMU	20110421	20100915	0.65	0.75 ↗	0.77	**0.83** ↗	0.76	0.80 ↗	0.78	0.82 ↗	0.73	0.79 ↗
CMU	20110421	20101221	0.36	0.56 ↑	0.54	0.63 ↗	0.49	0.48 →	0.59	0.62 ↗	0.64	**0.67** ↗
CMU	20110421	20110202	0.39	0.42 ↗	0.62	0.72 ↗	0.49	0.47 →	0.64	0.69 ↗	0.72	**0.73** →
Nordland1000	spring	winter	0.25	0.46 ↑	0.02	0.06 ↑	0.06	0.10 ↑	0.07	0.11 ↑	0.77	**0.99** ↑
Nordland1000	spring	summer	0.66	**0.99** ↑	0.20	0.76 ↑	0.43	0.89 ↑	0.45	0.88 ↑	0.74	**0.99** ↑
Nordland1000	summer	winter	0.57	**0.95** ↑	0.05	0.20 ↑	0.05	0.15 ↑	0.16	0.53 ↑	0.45	0.94 ↑
Nordland1000	summer	fall	0.92	**1.00** ↗	0.53	0.98 ↑	0.82	0.98 ↗	0.79	0.97 ↗	0.90	**1.00** ↗
StLucia	100909_0845	180809_1545	0.46	**0.77** ↑	0.08	0.15 ↑	0.28	0.36 ↑	0.15	0.40 ↑	0.46	**0.77** ↑
StLucia	100909_1000	190809_1410	0.57	0.85 ↑	0.19	0.47 ↑	0.46	0.59 ↑	0.22	0.62 ↑	0.64	**0.86** ↑
StLucia	100909_1210	210809_1210	0.66	0.83 ↑	0.61	0.78 ↑	0.84	**0.85** →	0.63	0.76 ↗	0.70	0.81 ↗
worst case			0.12	0.27 ↑	0.02	0.06 ↑	0.05	0.05 ↗	0.07	0.11 ↑	0.45	**0.67** ↑
best case			0.92	**1.00** ↗	0.99	**1.00** →	0.98	**1.00** →	0.95	**1.00** →	0.91	**1.00** ↗
average case (mAP)			0.52	0.70 ↑	0.50	0.67 ↑	0.55	0.62 ↗	0.46	0.58 ↑	0.71	**0.86** ↗

1.2. Encoding local image feature position and scale

An example of additional image information that can be encoded is the scale of local image features. In the paper, eq. 4 encoded a set of local features L_i with pose encodings P_i. Here, we extend this with information about the image scale level λ_i at which a particular local features was detected. We create a scale encoding Λ_i of λ_i similar to the pose encoding described in the paper Sec. 3.1.5: We sample two random basis vectors Λ^{min}, Λ^{max} for the minimum and maximum scale levels and interpolate vectors in between. Since this encoding will be used together with the pose encoding, we want to limit possible interference with the systematic pose encoding by concatenation. Since there is only a small discrete set of used image scales, we compute a fixed interpolated vector Λ for each scale by sampling each dimension independently either from

Λ^{min} or Λ^{max} with probability proportional to the distance between the corresponding scale level and the minimum and maximum scales.

The holistic image encoding L is then computed very similar to eq. 4 in the paper:

$$L = \bigoplus_{i=1}^{k} (L_i \otimes P_i \otimes \Lambda_i) \tag{2}$$

Table 2 shows results when using this encoding in combination with DELF features from multiple scales. In summary, the HDC encoding of scales works, however, on these particular place recognition datasets, using features at multiple scales does not improve the overall performance.

We extract 200 DELF fetures at multiple scales (following [12] we use scales [0.25, 0.35, 0.5, 0.71, 1, 1.41, 2]). The comparison of the two *exhaustive* feature comparisons in columns 4 (single scale) and 5 (multiple scales) of Table 2 show a reduced performance when using DELF features from multiple scales. Presumably this is an effect of the severe appearance changes (which challenge repetitive feature detection) and the mild viewpoint changes (which reduces the importance of scale invariance). Also, when using the presented HDC encoding from eq. 2 to encode positions *and* scales of features, there is only very little additional benefit from the scale information (on these particular datasets); this can be seen by comparing the HDC encodings of pose only (column 6) with the additional encoding of scale (column 7). There is also one experiment where the performance considerably drops after adding scale information (the fifth Oxford comparison).

Presumably, the amount of information from the positions of local landmarks is much higher than that from scale and the latter is almost completely hidden. To evaluate whether the scale encoding works in principle, we reduce the importance of the position information by reducing the position weighting parameters to $n_x = 1$ and $n_y = 4$. Results are shows in columns 8 and 9 of Table 2. In this setup, the benefit of additionally encoding scale increases. However, the overall performance is lower compared to using optimized position parameters and single scale features, but we consider this a property of the datasets and not the proposed encoding (since the same behavior appears for exhaustive comparisons in columns 4 and 5).

Table 2: Influence of extracting features at multiple scales. In case of features at multiple scale levels, we can either include the scale level in the feature distance function (w scale) or not (w/o scale). The table reports average precision. Arrows compare to the column to the left (they indicate similar performance or more than ±5 % or ±25 % performance change).

Dataset	**DB**	**Query**	**DELF-pos single scale** (exhaustive)	**DELF-M-pos multiscale** (exhaustive)	**HDC-DELF-M multiscale** w/o scale	**HDC-DELF-M multiscale** w scale	**HDC-DELF-M n_x=1, n_y=4 multiscale** w/o scale	**HDC-DELF-M n_x=1, n_y=4 multiscale** w scale
GardensPointWalking	day_right	day_left	0.95	0.90 ↘	0.76	0.75 →	**0.90**	**0.90** →
	day_right	night_right	0.80	0.71 ↘	0.65	**0.68** →	0.54	0.58 ↗
	day_left	night_right	0.73	0.53 ↓	0.33	0.34 →	0.35	**0.40** ↗
OxfordRobotCar	2014-12-09-13-21-02	2015-05-19-14-06-38	0.97	0.91 ↘	**0.87**	**0.87** →	0.59	0.59 →
	2014-12-09-13-21-02	2015-08-28-09-50-22	0.60	0.66 ↗	0.57	**0.58** →	0.47	0.54 ↗
	2014-12-09-13-21-02	2014-11-25-09-18-32	0.88	0.91 →	**0.89**	0.88 →	0.83	0.86 →
	2014-12-09-13-21-02	2014-12-16-18-44-24	0.64	0.43 ↓	**0.18**	0.17 →	0.04	0.06 ↗
	2015-05-19-14-06-38	2015-02-03-08-45-10	0.92	0.94 →	**0.78**	0.70 ↘	0.49	0.41 ↘
	2015-08-28-09-50-22	2014-11-25-09-18-32	0.72	0.70 →	**0.55**	0.54 →	0.42	0.47 ↗
SFUMountain	dry	dusk	0.81	0.70 ↘	0.74	**0.75** →	0.70	0.70 →
	dry	jan	0.82	0.58 ↓	0.53	**0.58** ↗	0.40	0.43 ↗
	dry	wet	0.82	0.67 ↘	0.64	**0.66** →	0.50	0.54 ↗
CMU	20110421	20100901	0.82	0.81 →	**0.79**	**0.79** →	0.76	0.77 →
	20110421	20100915	0.77	0.75 →	0.69	0.71 →	0.71	**0.74** →
	20110421	20101221	0.67	0.65 →	**0.66**	0.65 →	0.61	0.61 →
	20110421	20110202	0.82	0.79 →	**0.76**	0.75 →	0.74	0.75 →
Nordland1k	spring	winter	0.86	0.79 ↘	0.64	**0.69** ↗	0.38	0.47 ↗
	spring	summer	0.71	0.63 ↘	0.63	**0.64** →	0.44	0.47 ↗
	summer	winter	0.44	0.42 →	0.47	**0.49** →	0.25	0.31 ↗
	summer	fall	0.91	0.80 ↘	0.82	**0.84** →	0.72	0.77 ↗
StLucia	100909_0845	180809_1545	0.45	0.40 ↘	0.34	**0.36** ↗	0.24	0.30 ↗
	100909_1000	190809_1410	0.64	0.55 ↘	0.52	**0.55** →	0.44	0.50 ↗
	100909_1210	210809_1210	0.68	0.61 ↘	0.53	0.58 ↗	0.53	**0.60** ↗
worst case			0.44	0.40 ↘	**0.18**	0.17 →	0.04	0.06 ↗
best case			0.97	0.94 →	0.89	0.88 →	**0.90**	**0.90** →
average case (mAP)			0.76	0.69 ↘	0.62	**0.63** →	0.52	0.55 ↗

1.3. How to switch to a different HDC architecture?

One reason why we consider HDC as a *framework* for feature aggregation is the opportunity to exchange the underlying HDC architecture. In the paper, we present an architecture based on the MAP [5] architecture using the real vector space $\mathbb{V} = \mathbb{R}^d$. Other established HDC architectures include binary spatter codes [7] or holographic reduced representations [13] that work on a complex vector space, please refer to [14] for a survey. The following steps are required in order to replace the underlying HDC architecture:

1. Replace the binding $\otimes$ and bundling operators $\oplus$ with the appropriate implementations.
2. Adapt the similarity metric to the new vector space (e.g. based on Hamming distance for binary vectors) as well as the way how random vectors are created.
3. Provide an appropriate image descriptor encoding in the underlying vectors space $\mathbb{V}$. The survey [14] also provides (add-hoc) examples how this can be done. Presumably, any information preserving transformation to a distributed representation in the appropriate vector space is a potential candidate. However, this is an important part of future work.
4. If required, adapt the way how additional information is encoded. E.g., the pose encoding from the paper is designed to work with different HDC architectures. However, for particular vector spaces, there are presumably better encodings available (e.g. using Fourier transform for complex vectors spaces as was done in [8].)

2. Detailed experimental results that were summarized in the paper

2.1. Unordered aggregation of holistic descriptors

The paper summarizes the results for the combination of multiple descriptors in boxplots. Tables 3 and 4 provide the individual results for each dataset comparison. It can be seen that both methods provide similar results. Although there are also cases where the performance decreases compared to individual descriptors (indicated by red and yellow arrows), for most cases, there is considerable improvement (green arrows).

Table 3: Holistic feature aggregation using HDC bundling. The table reports average precision. Arrows compare to the individual holistic descriptors in the order of appearance in the name (they indicate similar performance or more than ± 5 % or ± 25 % performance change)

Dataset	DB	Query	AN	NV	DV	AN+NV	NV+DV	AN+DV	AN+NV+DV
GardensPointWalking	day_right	day_left	0.51	0.98	0.98	0.91 ↑ ↘	**0.99** → →	0.91 ↑ ↘	0.97 ↑ → →
	day_right	night_right	0.58	0.53	0.40	**0.70** ↗ ↑	0.57 ↗ ↑	0.63 ↗ ↑	0.69 ↗ ↑ ↑
	day_left	night_right	0.18	0.37	0.16	**0.43** ↑ ↗	0.32 ↘ ↑	0.22 ↗ ↑	0.39 ↑ → ↑
OxfordRobotCar	2014-12-09-13-21-02	2015-05-19-14-06-38	0.76	0.86	0.87	**0.95** ↗ ↗	0.87 → →	0.94 ↗ ↗	**0.95** ↑ ↗ ↗
	2014-12-09-13-21-02	2015-08-28-09-50-22	0.40	0.63	0.54	0.70 ↑ ↗	0.70 ↗ ↑	0.66 ↑ ↗	**0.75** ↑ ↗ ↑
	2014-12-09-13-21-02	2014-11-25-09-18-32	0.69	**0.92**	0.90	0.91 ↑ →	**0.92** → →	0.89 ↑ →	**0.92** ↑ → →
	2014-12-09-13-21-02	2014-12-16-18-44-24	0.27	0.08	0.07	**0.36** ↑ ↑	0.11 ↑ ↑	0.19 ↓ ↑	0.23 ↘ ↑ ↑
	2015-05-19-14-06-38	2015-02-03-08-45-10	0.82	0.94	0.27	**0.96** ↗ →	0.88 ↘ ↑	0.73 ↘ ↑	0.92 ↗ → ↑
	2015-08-28-09-50-22	2014-11-25-09-18-32	0.33	0.57	0.42	**0.60** ↑ ↗	0.55 → ↑	0.50 ↑ ↗	0.59 ↑ → ↑
SFUMountain	dry	dusk	0.54	0.46	**0.76**	0.63 ↗ ↑	0.74 ↑ →	0.73 ↑ →	0.75 ↑ ↑ →
	dry	jan	0.36	0.21	**0.58**	0.45 ↗ ↑	0.48 ↑ ↘	0.54 ↑ ↘	0.57 ↑ ↑ →
	dry	wet	0.43	0.38	**0.72**	0.55 ↑ ↑	0.71 ↑ →	0.67 ↑ ↘	**0.72** ↑ ↑ →
CMU	20110421	20100901	0.52	0.66	0.64	0.70 ↑ ↗	0.72 ↗ ↗	0.67 ↑ →	**0.73** ↑ ↗ ↗
	20110421	20100915	0.67	0.76	0.73	0.76 ↗ →	**0.78** → ↗	0.75 ↗ →	**0.78** ↗ → ↗
	20110421	20101221	0.44	0.51	0.47	**0.57** ↑ ↗	0.46 ↘ →	0.51 ↗ ↗	0.56 ↑ ↗ ↗
	20110421	20110202	0.43	0.56	0.45	**0.61** ↑ ↗	0.55 → ↗	0.50 ↗ ↗	0.57 ↑ → ↑
Nordland1k	spring	winter	**0.29**	0.02	0.06	0.11 ↓ ↑	0.04 ↑ ↓	0.19 ↓ ↑	0.10 ↓ ↑ ↑
	spring	summer	**0.72**	0.19	0.42	0.61 ↘ ↑	0.46 ↑ ↗	0.69 → ↑	0.66 ↘ ↑ ↑
	summer	winter	**0.54**	0.04	0.04	0.32 ↓ ↑	0.06 ↑ ↑	0.36 ↓ ↑	0.26 ↓ ↑ ↑
	summer	fall	**0.92**	0.52	0.81	0.87 → ↑	0.81 ↑ →	**0.92** → ↗	0.91 → ↑ ↗
StLucia	100909_0845	180809_1545	0.45	0.07	0.23	0.36 ↘ ↑	0.21 ↑ ↘	**0.48** → ↑	0.41 ↘ ↑ ↑
	100909_1000	190809_1410	0.58	0.18	0.38	0.56 → ↑	0.40 ↑ →	**0.62** ↗ ↑	0.60 → ↑ ↑
	100909_1210	210809_1210	0.66	0.58	**0.82**	0.72 ↗ ↗	0.81 ↑ →	0.81 ↗ →	0.81 ↗ ↑ →
worst case			0.18	0.02	0.04	0.11 ↓ ↑	0.04 ↑ →	**0.19** → ↑	0.10 ↓ ↑ ↑
best case			0.92	0.98	0.98	0.96 → →	**0.99** → →	0.94 → →	0.97 ↗ → →
average case (mAP)			0.53	0.48	0.51	0.62 ↗ ↑	0.57 ↗ ↗	0.61 ↗ ↗	**0.64** ↗ ↑ ↑

Table 4: These results correspond to Table 3 but were created using concatenation of descriptors instead of bundling. They are very similar.

Dataset	DB	Query	AN	NV	DV	AN+NV	NV+DV	AN+DV	AN+NV+DV
GardensPointWalking	day_right	day_left	0.51	0.98	0.98	0.91 ↑ ↘	**0.99** → →	0.90 ↑ ↘	0.96 ↑ → →
	day_right	night_right	0.58	0.53	0.40	0.67 ↗ ↑	0.57 ↗ ↑	0.64 ↗ ↑	**0.69** ↗ ↑ ↑
	day_left	night_right	0.18	0.37	0.16	**0.43** ↑ ↗	0.34 ↘ ↑	0.21 ↗ ↑	0.37 ↑ → ↑
OxfordRobotCar	2014-12-09-13-21-02	2015-05-19-14-06-38	0.76	0.86	0.87	**0.93** ↗ ↗	0.87 → →	0.91 ↗ ↗	**0.93** ↗ ↗ ↗
	2014-12-09-13-21-02	2015-08-28-09-50-22	0.40	0.63	0.54	0.68 ↑ ↗	0.68 ↗ ↑	0.63 ↑ ↗	**0.75** ↑ ↗ ↑
	2014-12-09-13-21-02	2014-11-25-09-18-32	0.69	**0.92**	0.90	**0.92** ↑ →	**0.92** → →	0.88 ↑ →	**0.92** ↑ → →
	2014-12-09-13-21-02	2014-12-16-18-44-24	0.27	0.08	0.07	**0.28** → ↑	0.10 ↗ ↑	0.19 ↓ ↑	0.20 ↘ ↑ ↑
	2015-05-19-14-06-38	2015-02-03-08-45-10	0.82	0.94	0.27	**0.95** ↗ →	0.86 ↘ ↑	0.81 → ↑	0.94 ↗ → ↑
	2015-08-28-09-50-22	2014-11-25-09-18-32	0.33	0.57	0.42	**0.63** ↑ ↗	0.57 → ↑	0.50 ↑ ↗	0.61 ↑ ↗ ↑
SFUMountain	dry	dusk	0.54	0.46	**0.76**	0.63 ↗ ↑	0.74 ↑ →	0.71 ↑ ↘	0.75 ↑ ↑ →
	dry	jan	0.36	0.21	**0.58**	0.42 ↗ ↑	0.50 ↑ ↘	0.53 ↑ ↘	0.55 ↑ ↑ →
	dry	wet	0.43	0.38	**0.72**	0.55 ↑ ↑	**0.72** ↑ →	0.66 ↑ ↘	0.71 ↑ ↑ →
CMU	20110421	20100901	0.52	0.66	0.64	0.69 ↑ →	**0.73** ↗ ↗	0.67 ↑ →	**0.73** ↑ ↗ ↗
	20110421	20100915	0.67	0.76	0.73	0.77 ↗ →	**0.79** → ↗	0.75 ↗ →	0.78 ↗ → ↗
	20110421	20101221	0.44	0.51	0.47	**0.55** ↗ ↗	0.46 ↘ →	0.49 ↗ →	0.51 ↗ → ↗
	20110421	20110202	0.43	0.56	0.45	**0.57** ↑ →	0.54 → ↗	0.50 ↗ ↗	0.55 ↑ → ↗
Nordland1k	spring	winter	**0.29**	0.02	0.06	0.11 ↓ ↑	0.04 ↑ ↓	0.18 ↓ ↑	0.10 ↓ ↑ ↑
	spring	summer	**0.72**	0.19	0.42	0.62 ↘ ↑	0.46 ↑ ↗	0.69 → ↑	0.66 ↘ ↑ ↑
	summer	winter	**0.54**	0.04	0.04	0.32 ↓ ↑	0.06 ↑ ↑	0.36 ↓ ↑	0.26 ↓ ↑ ↑
	summer	fall	**0.92**	0.52	0.81	0.87 → ↑	0.82 ↑ →	**0.92** → ↗	0.91 → ↑ ↗
StLucia	100909_0845	180809_1545	0.45	0.07	0.23	0.37 ↘ ↑	0.20 ↑ ↘	**0.47** → ↑	0.40 ↘ ↑ ↑
	100909_1000	190809_1410	0.58	0.18	0.38	0.56 → ↑	0.40 ↑ →	**0.62** ↗ ↑	0.59 → ↑ ↑
	100909_1210	210809_1210	0.66	0.58	**0.82**	0.72 ↗ ↗	0.80 ↑ →	**0.82** ↗ →	0.81 ↗ ↑ →
worst case			**0.18**	0.02	0.04	0.11 ↓ ↑	0.04 ↑ →	**0.18** → ↑	0.10 ↓ ↑ ↑
best case			0.92	0.98	0.98	0.95 → →	**0.99** → →	0.92 → ↘	0.96 → → →
average case (mAP)			0.53	0.48	0.51	0.62 ↗ ↑	0.57 ↗ ↗	0.61 ↗ ↗	**0.64** ↗ ↑ ↑

2.2. Detailed results for parameters n_x and n_y

The paper demonstrates the influence of the spatial weighting parameters n_x and n_y on the GardensPointWalking dataset. Table 5 also shows results on all other datasets for all combinations of $n_x \in \{2, 3, 4\}$ and $n_y \in \{5, 6\}$. The parameters $n_x = 4, n_y = 6$ that are used in the paper provide the overall best results. However, down-weighting horizontal displacements (by decreasing n_x) can lead to considerably better results in case of horizontal viewpoint changes, e.g. on GardensPointWalking. According adjustments are particularly important for other image retrieval tasks with more severe viewpoint changes.

Table 5: Influence of parameters n_x and n_y for HDC-DELF. The table reports average precision.

Dataset	DB	Query	$n_x = 2, n_y = 5$	$n_x = 2, n_y = 6$	$n_x = 3, n_y = 5$	$n_x = 3, n_y = 6$	$n_x = 4, n_y = 5$	$n_x = 4, n_y = 6$
GardensPointWalking	day_right	day_left	0.90	**0.92**	0.86	0.86	0.81	0.82
GardensPointWalking	day_right	night_right	0.79	**0.80**	**0.80**	**0.80**	0.78	0.79
GardensPointWalking	day_left	night_right	0.58	**0.60**	0.56	0.54	0.51	0.47
OxfordRobotCar	2014-12-09-13-21-02	2015-05-19-14-06-38	0.85	0.86	0.89	0.89	**0.91**	**0.91**
OxfordRobotCar	2014-12-09-13-21-02	2015-08-28-09-50-22	0.58	0.65	0.64	0.62	0.69	**0.71**
OxfordRobotCar	2014-12-09-13-21-02	2014-11-25-09-18-32	0.76	0.77	0.78	0.77	**0.83**	0.82
OxfordRobotCar	2014-12-09-13-21-02	2014-12-16-18-44-24	0.59	0.70	0.69	0.75	0.74	**0.80**
OxfordRobotCar	2015-05-19-14-06-38	2015-02-03-08-45-10	0.74	0.75	0.69	0.74	0.77	**0.78**
OxfordRobotCar	2015-08-28-09-50-22	2014-11-25-09-18-32	0.59	0.64	0.64	0.65	0.66	**0.71**
SFUMountain	dry	dusk	0.75	0.74	0.78	0.78	0.79	**0.81**
SFUMountain	dry	jan	0.41	0.40	0.45	0.49	0.56	**0.57**
SFUMountain	dry	wet	0.66	0.64	0.72	0.71	0.74	**0.75**
CMU	20110421	20100901	0.73	0.73	0.75	0.75	**0.76**	0.75
CMU	20110421	20100915	**0.75**	0.74	0.74	0.74	0.74	0.73
CMU	20110421	20101221	0.61	0.62	0.62	**0.64**	0.63	**0.64**
CMU	20110421	20110202	0.71	0.71	0.71	**0.73**	**0.73**	0.72
Nordland1000	spring	winter	0.58	0.61	0.68	0.70	0.73	**0.77**
Nordland1000	spring	summer	0.62	0.63	0.69	0.69	0.72	**0.74**
Nordland1000	summer	winter	0.29	0.31	0.37	0.39	0.40	**0.45**
Nordland1000	summer	fall	0.84	0.84	0.88	0.88	0.89	**0.90**
StLucia	100909_0845	180809_1545	0.36	0.36	0.41	0.42	0.44	**0.46**
StLucia	100909_1000	190809_1410	0.56	0.57	0.61	0.62	0.63	**0.64**
StLucia	100909_1210	210809_1210	0.69	0.69	**0.70**	0.69	0.69	**0.70**
worst case			0.29	0.31	0.37	0.39	0.40	**0.45**
best case			0.90	**0.92**	0.89	0.89	0.91	0.91
average case (mAP)			0.65	0.66	0.68	0.69	0.70	**0.71**

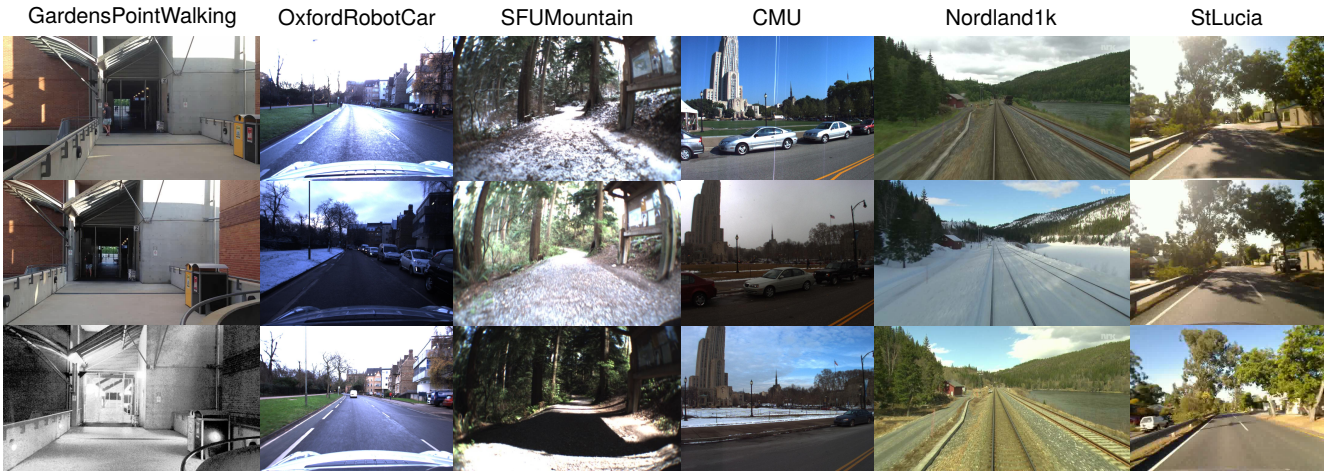

Figure 1: Example images from the used standard mobile robotics place recognition datasets.

3. Dataset details

All evaluations are based on six different datasets with different characteristics regarding the environment, appearance changes, in-sequence loops, stops, or viewpoint changes. Example images of same places under different conditions for all datasets are shown in Fig. 1. **GardensPointWalking** [1]: 600 images, recorded with a hand-held camera on a single route on campus, twice during daytime on the left and right side of a pathway and once at night on the right side. **OxfordRobotCar** [9]: about 13k images, car rides collected over one year under several conditions. We use the center camera of the trinocular stereo camera. **SFU** Mountain [4]: about 1.5k images, multiple traverses of a mobile robot in semi-structured woodland under varying lighting and weather conditions. **CMU** Visual Localization [3]: about 5.6k images, five car rides along a 8km route with stops, weather, and seasonal changes. We use the left camera. **Nordland** [2]: 4k images, viewpoint aligned rides along a train track once per season. We use a subset of 1000 distinct places (without tunnels). **StLucia** (Various Times of the Day) [6]: about 8k images, collected with a webcam mounted on a car driving in a suburb from morning to afternoon over several days.

References

[1] *A. Glover. "Day and Night with Lateral Pose Change Datasets". (2014).* https://goo.gl/tqmWyq. 6

[2] *Norwegian Broadcasting Corporation "Nordlandsbanen - Minutt for Minutt".* http://nrkbeta.no/2013/01/15/nordlandsbanen-minute-by-minute-season-by-season/. 6

[3] H. Badino, D. Huber, and T. Kanade. Visual topometric localization. In *Intelligent Vehicles Symposium (IV)*, 2011. 6

[4] Jake Bruce, Jens Wawerla, and Richard Vaughan. The SFU mountain dataset: Semi-structured woodland trails under changing environmental conditions. In *ICRA Workshop on Visual Place Rec. in Changing Environments*, 2015. 6

[5] Ross W. Gayler. Vector Symbolic Architectures answer Jackendoff's challenges for cognitive neuroscience. In *Int. Conf. on Cognitive Science*, 2003. 4

[6] A. J. Glover, W. P. Maddern, M. J. Milford, and G. F. Wyeth. Fab-map + ratslam: Appearance-based slam for multiple times of day. In *Int. Conf. on Robotics and Automation (ICRA)*, 2010. 6

[7] Pentti Kanerva. Hyperdimensional Computing: An Introduction to Computing in Distributed Representation with High-Dimensional Random Vectors. *Cognitive Computation*, 1(2):139–159, 2009. 4

[8] Brent Komer, Terrence C. Stewart, Aaron Voelker, and Chris Eliasmith. A neural representation of continuous space using fractional binding. In *CogSci*, pages 2038–2043, 2019. 4

[9] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The oxford robotcar dataset. *The Int. Journal of Robotics Research*, 36(1):3–15, 2017. 6

[10] M. Milford and G. F. Wyeth. SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012. 1, 2

[11] Peer Neubert, Stefan Schubert, and Peter Protzel. An introduction to hyperdimensional computing for robotics. *Künstliche Intell.*, 33(4):319–330, 2019. 1

[12] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han. Large-scale image retrieval with attentive deep local features. In *Int. Conf. on Computer Vision (ICCV)*, 2017. 3

[13] Tony Plate. *Distributed Representations and Nested Compositional Structure*. PhD thesis, 1994. 4

[14] Kenny Schlegel, Peer Neubert, and Peter Protzel. A comparison of vector symbolic architectures. *CoRR*, abs/2001.11797, 2020. 4