Topological localization with omnidirectional images using an Echo State Network

(Bachelor thesis)

Michaja Broertjes, Sjoerd de Jong

August 12, 2009

Abstract

We present a model for topological localization of robots equipped with hyperbolic mirrors for omnidirectional vision. Our goal is to achieve localization on a biologically plausible manner. The omnidirectional images are first pre-processed to data which can be used as input for an artificial neural network. The model uses an Echo State Network to perform a localization based on current input and past network activation. We test our model using a dataset of annotated omnidirectional images. We compare the results of our model with two baseline algorithms, namely a nearest neighbour algorithm and a Multi Layer Perceptron and with other research performed on the same dataset. We also show that a simple step in the pre-processing stage, which is inspired by SIFT and uses gradients, greatly improves the performance of the model.

1 Introduction

Early research on robotics focused on making a model of the world on which robots could base their actions. This approach failed because it was infeasible to model the whole environment a robot is in. The resulting robots using this approach could only function within simple experimental environments. Later another approach became standard, where robot action is controlled by behaviours. A behaviour consists of a connection between sensory input and motor output. Robots using this reactive paradigm are responsive to dynamic environments and show intelligent action resembling that of insects. Nevertheless, such robots are not able to plan their actions; therefore a model of the world is necessary. For an extensive discussion on this subject see Brooks (1991). As of today, successful approaches combine both low level reactive actions and higher level actions based on map building and path planning. Our research focuses on localization, important for robots because path planning or navigation always starts with localization. Our goal is to design a localization model that is as biologically plausible as possible.

Researchers have found various ways to effectively create maps for localization and navigation while coping with complex worlds. Robots need sensors to perceive their environments and detect properties in their environment which are recognizable and distinctive. Robots which use vision as primary sensory input need to detect either distinctive image features or landmarks. Based on these image features or landmarks the robot is able to recognize places and build maps for localization and navigation. Some researchers studied ways to efficiently and accurately detect these image features or landmarks. Lowe (2004) introduced a wide-baseline-matching technique namely the scale invariant features (SIFT) extractor, which is broadly used in the robotics field. We do not use SIFT because SIFT produces a varying number of features to describe an image and our model uses an artificial neural network which needs a fixed number of inputs. Another reason we choose to not use SIFT is that SIFT is designed for normal images and we use omnidirectional images. Murillo, Sagüés, Guerrero, Goedeme, Tuytelaars, and Van Gool (2007) proposed a features extractor for omnidirectional images based on polar lines. In comparison While reducing computational load, it keeps a high accuracy in comparison to SIFT. Unfortunately this

feature extractor also produces a varying number of features. They also showed how a hierarchical matching of images can overcome the problems with large image sets. Matching of images by image features can be used for localization as well as navigation. Localization is done by keeping images from visited places in a visual memory and comparing this images with the images retrieved from the current localization. For navigation it is necessary for the robot to also memorize which action or actions took the robot from one place to another. Memorized or executed actions are liable to small errors caused by noise. If a robot just executes a set of memorized actions these errors can accumulate to a large error. Matching of perceived and memorized images can be used to correct for these mistakes and keep the robot on track.

Other researchers are more interested in building and representing maps of the environment instead of just matching images with each other. Typically there are two sorts of maps: metric maps and topological maps. Metric maps are very useful for planning actions, because they hold precise information of the robot and locations of other objects. The major downside of these metric maps is that they are hard to compute for larger environments. Topological maps only hold information about places and connections between them and are better suited for larger environments. Places are areas that are abstracted to one node on the map, for example rooms in a building.

Combinations of metrical and topological maps are also possible. Blanco, González, and Fernández-Madrigal (2009) showed a study where the nodes of a topological map hold a local metric map. This approach efficiently copes with larger environments by using the scalability of topological maps and the accuracy of local metric maps; it shows that a hybrid solution with both map types can combine the strengths of both map types. Kuipers (2000) proposes the spatial semantic hierarchy (SSH), a hierarchy of interacting levels with among others a metrical and a topological level.

As our goal is to achieve topological localization as biologically plausible as possible our model uses a biologically inspired artificial neural network. Rizzi and Cassinis (2001) showed a method to process omnidirectional images to data in such a way that a neural network can take the processed data as input. They used a conical mirror with a non-optimal reflecting surface resulting in a blurred image. This blurred image only consists of visual aspects that appear larger in the image while the smaller details are thrown away. This is useful since the larger visual aspects are more likely to be consistent over time than smaller aspects.

In contrast to Rizzi and Cassinis (2001), our model uses an Echo State Network (ESN) (Jaeger, 2002) instead of a normal feedforward neural network. Feedforward neural networks can be trained to recognize different patterns in the data, but are not able to use earlier input to classify current input. This is the main problem with neural modelling, creating a model that can process a continuous stream of input. Some solutions have been found but these models are most often task dependent. ESN in contrary can be used for real-time computing on time-varying input.

We use a dataset(Zivkovic, Booij, and Kröse, 2007) of omnidirectional images taken by a robot which autonomously drove around in an indoor environment. We will show that in comparison with two baseline algorithms and another approach (Murillo et al., 2007) tested on the same dataset our model is capable of accurate localization.

In the next section we will discuss the proposed model, followed by a description of our implementation and experiments, then we present results of experiments with the dataset and finally we draw our conclusions and end with a discussion.

2 Model

The model is roughly built up in two stages. The first stage is the processing of images into data that can serve as input for the neural network. The main idea for this image processing is to divide the hyperbolic image in parts of ten degrees and calculate the average colour value for every part, inspired by Rizzi and Cassinis (2001). This process is described in the first subsection. Next the model uses this data together with odometry data as input for an ESN. The output generated by this ESN leads to a localization. The ESN is described in the second subsection.

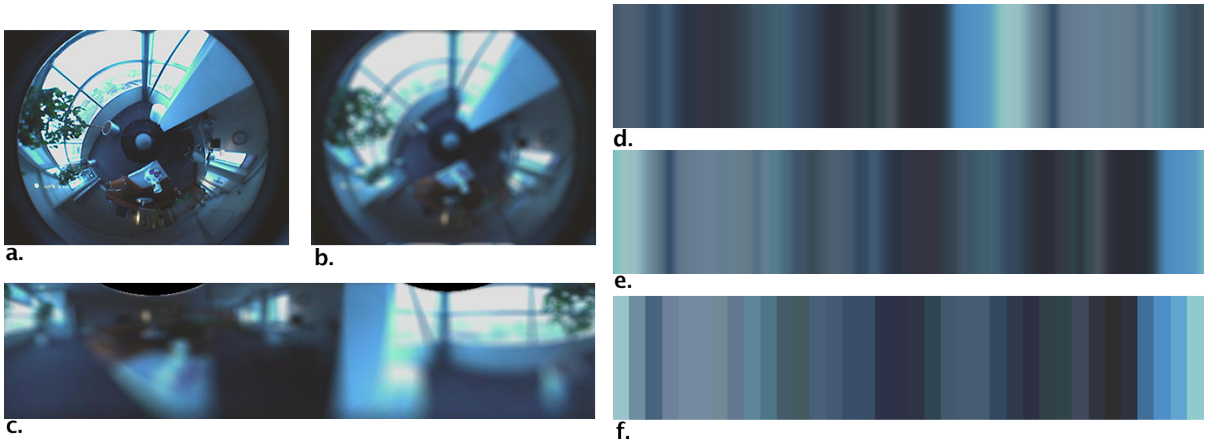

Figure 1: The original hyperbolic image (**a.**) is first blurred (**b.**), then transformed to a panoramic image (**c.**), then vertically averaged (**d.**), then normalized by intensity (**e.**) and finally divided in 36 pieces with the average colour value (**f.**).

2.1 Image processing

First of all the hyperbolic images (Figure 1a) are blurred (Figure 1b), this blurring acts as a low-pass filter: objects that appear to be small or far away are filtered out of the image. The remaining objects are objects that appear to be large or nearby, like doors, walls and closets. These objects tend to be more consistent over time, since smaller objects are easily and more often moved.

Next the model transforms the hyperbolic images to panoramic images (Figure 1c) to simplify further calculations. We used the Matlab code that came along with the dataset to do this transformation. Artifacts of this transformation are two black 'half moons' in the top of the image. As we wish that the model recognizes a location independently of its orientation we decided to cut off the upper part of the image. Next for every column the average colour value is calculated (Figure 1d). Hafner (2000) used a similar approach in her model, which also uses a neural network.

The images are normalized in such a way that the part of the image with the highest intensity is moved to the left side of the image (Figure 1e). This is done in the following manner. The algorithm finds blocks of adjacent pixels that are more than 85% of the maximal intensity. The minimum size of these blocks is 1/36th of the image. If a found block is smaller than this minimum size then pixels to the right of the block are included to retrieve a block with the minimum size. Then the model calculates the average colour value for all the found blocks. The block with the highest average colour value is moved to $x = 0$.

The model then divides the image into 36 segments, all representing ten degrees in the hyperbolic image (Figure 1f). For every segment the average colour value is calculated, resulting in 36 RGB values. The three channels of the RGB values are saved in a vector with a total length of 108.

Finally the model executes another normalization using gradients, inspired by the normalization used in the SIFT algorithm. The gradient normalization in the SIFT algorithm was inspired by a model of biological vision which also uses gradients in vision. Our gradient normalization transforms the 108 elements long vector into a vector with gradients. This is done by subtracting every value of its following value, with the exception of the last value which is subtracted of the first value. Next the vector is normalized to the size of the unit vector and the elements are moved in such a way that the largest value becomes the first value. In our experiments we run our model with and without this gradient normalization step to show its importance.

2.2 Echo State Network

Echo State Networks (Jaeger, 2002) are a special type of recurrent neural networks and are similar to Liquid State Machines (Maass, Natschlager, and

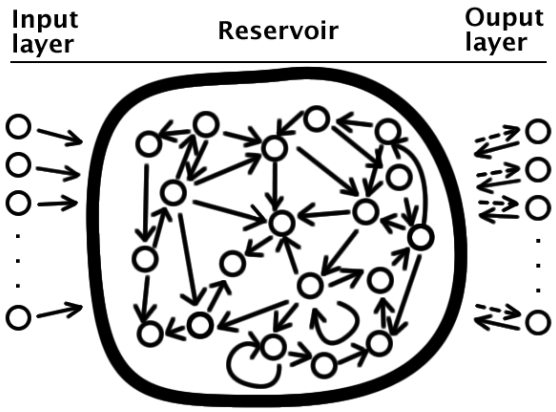

Figure 2: A schematic view of an ESN. At the left side a set of input neurons, in the middle a large random fixed reservoir of recurrent neurons and at the right side the output neurons with feedback connections. Solid arrows indicate fixed connections, dashed arrows indicate connections changed during training.

Markram, 2002). Both neural networks are known as reservoir computing. The main idea of reservoir computing is to feed a large random fixed recurrent neural network, the reservoir, with the input signal. The neurons in the reservoir respond non linear to the input signal. A set of simple linear output neurons are connected to the reservoir. The weights of these neurons can be trained by simple online or offline learning rules to linear combine the different activations of the neurons in the reservoir to the desired output. The network has feedback connections from the output neurons back to the neurons in the reservoir. Because of these feedback connections and the recurrent reservoir the network activation can be maintained while the network is fed new input. This property is not shared with feed forward neural networks and is the reason why ESN can handle temporal input signals. The only connections that are changed during learning are the connections between the reservoir and the output neurons. All the other connections are randomly initiated and fixed (Figure 2). During training the activation of the output neurons is set to the desired output, during testing the output neurons are activated by the reservoir and the reservoir in his turn is activated by the output neurons through the feedback connections.

The ESN used in our model has 110 input values, next to the 108 values resulting from the image processing (see §2.1) two odometry values are used: the translational speed v and the rotational speed ω. We decided to use a dynamic reservoir with a size of 440 neurons, four times the number of input values, large enough to produce enough non linear reactions to the input data for the output neurons to linear combine to the desired output. The number of output neurons depends on the number of places in the dataset that have to be learned. For every place our model has an output neuron that during training is set to 1 if a particular image was taken in that place.

The ESN has a set of parameters that influence the properties of the network. Those parameters are: the connection probability between the input neurons and the neurons in the reservoir (C_{in}), the connection probability between the neurons in the reservoir (C), the connection probability between the output neurons and the neurons in the reservoir (C_{back}) and the spectral radius (α). The three connection probabilities determine how much connections are made in the ESN during initialization. High connection probabilities result in a network rich of connections and low connection probabilities result in a less dense network. The random initiated weight matrices are scaled to unit size and then multiplied by the spectral radius. It thus influences the decay of network activation and the influence of past input on current input. A high α makes that the activation in the network is slowly decaying while in a network with a low α the activation decays fast.

3 Method

The dataset that we use for our experiments was created by Zivkovic et al. (2007). In the first subparagraph we describe the dataset and our modifications to it. The next subparagraph explains which method we used to find optimal parameters for our ESN. The last subparagraph shows the experiments we will do and what baseline algorithms we will use. All the discussed algorithms are implemented in Matlab.

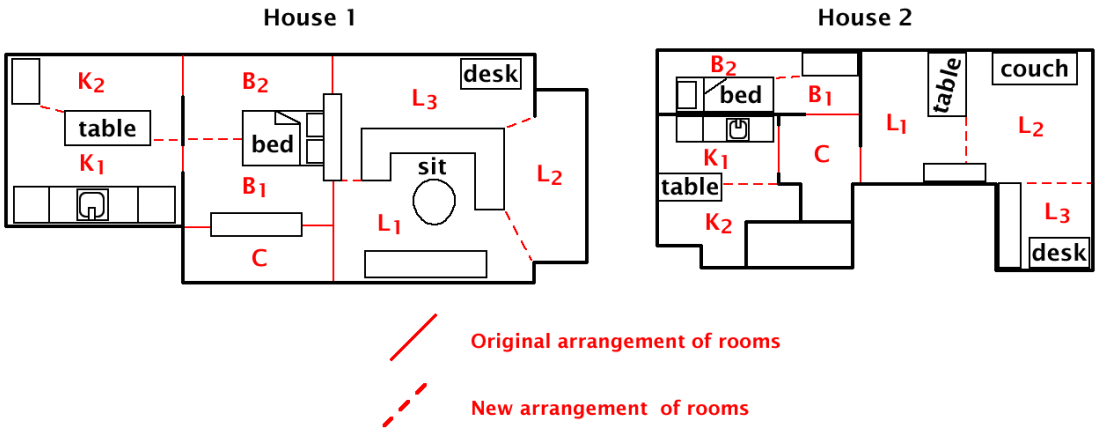

Figure 3: The new arrangement of places.

3.1 Dataset

The dataset is made up of images taken by a robot with omnidirectional vision. A camera was pointed vertically to a hyperbolic mirror giving a complete 360 degree view of the robots surroundings. The robot was driven three times through two different normal houses (the runs were numbered 1, 4 and 5). The runs in house 1 resulted in around 2000 images for a complete run, while house 2 yielded around 1600 images for a complete run. Unfortunately only the odd numbered images were annotated with places, so we decided to annotate every even numbered image with the annotation of its preceding odd numbered image. Both houses had four places: a corridor, a living room, a kitchen and a bedroom. We added a vector with four binary values to the dataset, indicating in which place the image was taken. We also pre-computed the 108 elements long vector as described in §2.1.

With every image and every odometry data point a timestamp was supplied. The supplied odometry data consisted out of a location (x, y) and an orientation θ relative to the start location and orientation of the robot. The timestamps of the images did not match the timestamps of the odometry data, so we linearly interpolated the odometry data with the images using the timestamps. Next we calculated the translational speed v and the rotational speed ω by dividing the distance and difference in orientation between two subsequent images by the time between the images.

To expand our experiments we rearranged the houses in eight places instead of the original four places. This makes the localization task a bit harder and also a bit more interesting. We divided the living room in three places, the kitchen and bedroom were divided in two places (Figure 3). For this new arrangement we also corrected the errors in the annotation of a series of images in run 4 of house 2.

We divided our dataset in training and test sets by taking one of the three runs as a training set and the other two runs as test sets. This can be done for all the three runs giving us six different combinations of train and test sets for both the houses.

3.2 Parameter optimization

To find the optimal parameters for our ESN we vary the following four parameters: C_{in}, C, C_{back} and α. We choose to vary the three connection probability parameters between the values 0.1, 0.5 and 0.9 because both high as low values for these parameters can result in a good working ESN. When the connection probability in the network is low, a specialized network can be trained with just the right connections. A high connection probability in contrary will enable the network to get the right connections along the bunch of other connections. These other connections can however negatively influence the network output. The spectral radius α must be between 0 and 1 and the influence of α is exponential. Therefore we decide to vary α between 0.98, 0.86 and 0.74. The combination of parameter

values that results in the highest average performance will be chosen as the optimal parameters.

3.3 Experiment

We will compare the performance of our model with the performance, the average percentage correctly classified images, of the algorithm used in Murillo et al. (2007). They proposed an algorithm that uses the gradients of polar lines in the hyperbolic image as descriptors of an image and then matches images in the test set with images in the training set. Because they used a very different method to test their model it is not possible to do exhaustive comparisons between the two algorithms. Therefore we will also compare our model with two baseline algorithms, a nearest neighbour algorithm and a Multi Layer Perceptron (MLP). These algorithms are described in §3.3.1 and §3.3.2. We will do this for four different test conditions as described in §3.3.3.

3.3.1 Nearest neighbour

The first baseline algorithm is a simple nearest neighbour algorithm which uses the Euclidean distance as a measure to match two vectors. For every image in the test set the algorithm finds the best matching image in the training set. The place in which this image was taken forms the localization of the image in the test set. Since this algorithm, other than our ESN and the MLP, is deterministic we only have to run the algorithm once for every test and training set.

3.3.2 Multi Layer Perceptron

The second baseline algorithm is a Multi Layer Perceptron (MLP) implemented with the Neural Network Toolbox of Matlab. The MLP has an input layer of 108 neurons, a hidden layer of 10 neurons and an output layer of either 4 or 8 neurons, depending on the number of places. The MLP is trained on a test set during a maximum of 50 epochs, it stops when it reaches a close-to-zero gradient. After training the MLP is fed with the test set and the most active output neuron determines the localization. The MLP is run 5 times for every combination of training and test set in all the four situations and the performance is averaged over every five runs.

Table 1: Optimal network parameters

house	test condition	C_{in}	C	C_{back}	α
1	basic model	0.5	0.5	0.5	0.86
2	basic model	0.5	0.1	0.5	0.86
1	gradient	0.5	0.5	0.1	0.98
2	gradient	0.9	0.5	0.1	0.98
1	$\frac{1}{50}$th images	0.9	0.5	0.1	0.98
2	$\frac{1}{50}$th images	0.5	0.5	0.1	0.86
1	extra places	0.9	0.9	0.1	0.98
2	extra places	0.9	0.1	0.1	0.98

3.3.3 Test conditions

In the first test condition we use the basic version of the model which does not execute the gradient localization inspired by SIFT. In the second and the other two test conditions the complete model is tested, including the gradient localization. In the third test condition the model is trained with $\frac{1}{50}$th of the images. The dataset with extra places per room is tested in the fourth test condition. We run the model 5 times for the six combinations of training and test sets in both houses. Because Murillo et al. (2007) used a test set with $\frac{1}{50}$th of the images we will use the third test condition to compare the results of both models.

4 Results

We first show the found optimal parameters for the four test conditions, these parameters are used in the experiments. Next we show the effect that the gradient normalization has on the performance. In the third subparagraph we compare our model with the two baseline algorithms, nearest neighbour algorithm and Multi Layer Perceptron. Then we compare the results of our model with the results of the algorithm described in Murillo et al. (2007). In the last subparagraph the results of experiments with extra places are presented. A complete view of the data for every combination of training and test set can be found in appendix A. Table 4 and table 5 show the average performance of the baseline algorithms and our model.

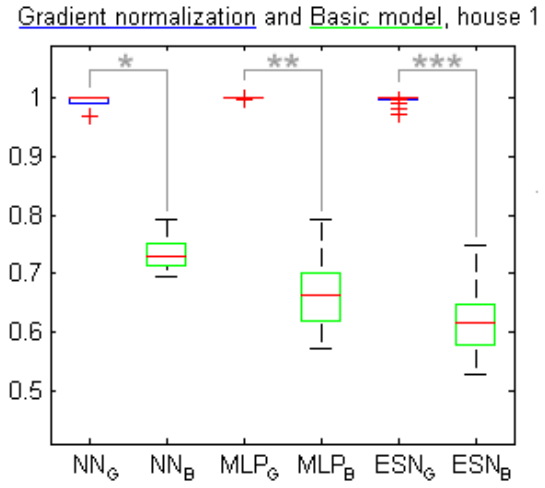

Figure 4: Comparison between gradient normalization and basic model for house 1, P-values with the Friedman test: $*P = 2.0 \cdot 10^{-13}$, $**P = 1.9 \cdot 10^{-11}$, $***P = 1.1 \cdot 10^{-10}$

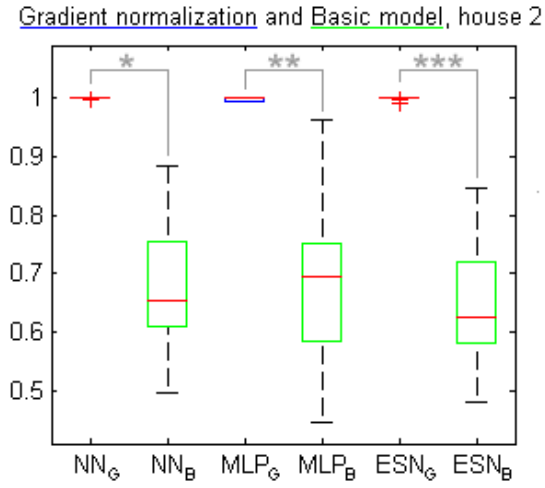

Figure 5: Comparison between gradient normalization and basic model for house 2, P-values with the Friedman test: $*P = 2.0 \cdot 10^{-13}$, $**P = 3.2 \cdot 10^{-11}$, $***P = 4.0 \cdot 10^{-11}$

4.1 Optimal parameters

For every test condition we have found the optimal parameters for our ESN, as described in §3.2 (see Table 1).

4.2 Gradient normalization

We have compared the results of the basic model with the model which uses the gradient normalization. Figure 4 and figure 5 show box plots of the results of the basic model and the model with gradient normalization for both houses. We used the five experiment results found for every combination of training and test set and for all the four test conditions (see §3.3). We used the Friedman test to determine that the performance of the model with gradient normalization is significant better than the basic model. The gradient normalization step improves the performance of as well our model as the two baseline algorithms with about 30%.

4.3 Baseline algorithms

To compare our ESN with the two baseline algorithms we used the Friedman test. For both houses we compared the results for every combination of training and test sets (see table 2 and table 3). The

Table 2: Friedman test on the scores of house 1

		P-values		
train	test	NN-MLP	NN-ESN	MLP-ESN
1	4	0.0051	0.5771	0.3450
1	5	0.0039	0.0050	0.0494
4	1	0.0038	0.0004	0.4440
4	5	$1.4 \cdot 10^{-6}$	$2.0 \cdot 10^{-7}$	0.1248
5	1	$1.4 \cdot 10^{-6}$	$2.3 \cdot 10^{-8}$	0.0555
5	4	0.0046	0.0053	0.0080

Friedman test was executed with the five results found for all the four test conditions (see §3.3). Values above the level of significance of 0.05 are coloured red. In most cases the nearest neighbour algorithm performs significantly better than the MLP and our ESN. If we compare the MLP and our ESN we see that in more than half of the cases our ESN does not score significantly different than the MLP.

4.4 $\frac{1}{50}$th images

Murillo et al. (2007) only produced results for house 1 and used a test set with $\frac{1}{50}$th of the images from the dataset. Therefore we included their results in table 4 under the $\frac{1}{50}$th images test condition. In

Table 3: Friedman test on the scores of house 2

		P-values		
train	test	NN-MLP	NN-ESN	MLP-ESN
1	4	$2.9 \cdot 10^{-5}$	$5.6 \cdot 10^{-7}$	0.3465
1	5	0.0001	0.5690	0.0039
4	1	0.0209	$7.8 \cdot 10^{-7}$	0.0104
4	5	0.0046	0.0232	0.0522
5	1	0.0257	0.0833	0.2219
5	4	0.1070	$3.0 \cdot 10^{-7}$	$1.2 \cdot 10^{-6}$

comparison with the results of the lines descriptor by Murillo et al. (2007) and with SIFT our model performs around 20%-30% better. If we however compare the performance of our model using $\frac{1}{50}$th images in house 1 (Table 4) with house 2 (Table 5), we suspect that the performance in house 1 may be a peak.

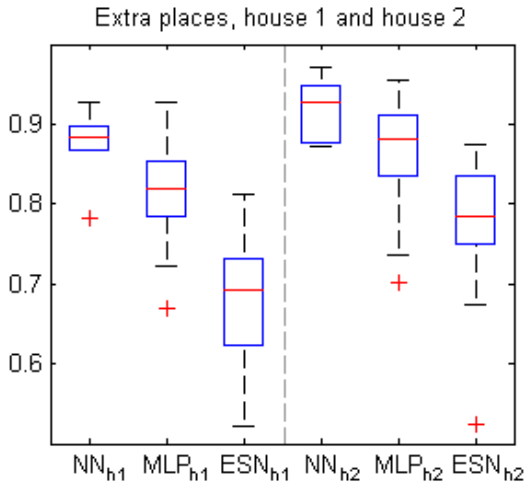

Figure 6: Performances with extra places for house 1 and house 2

4.5 Extra places

Figure 6 shows a box plot of the five results found for every training and test set combination in the multiple places per room setup for both the houses. The nearest neighbour algorithm classified about 10% less images correct, the MLP scored 10%-20% less and the MLP's performance dropped with 20%-30%.

5 Conclusion

If we compare the results of the basic model with the results of the model using the gradient normalization we see a major improvement of 25%-35%. By using the gradient normalization, contrasts in the images are magnified and plain surfaces are discarded. Because the vectors consist of gradients and are normalized to unit length, images taken in the same place under different illuminations result in the same vector. This makes our image descriptor a lot more robust.

Results with the new arrangement of the houses with extra places show that the performance of our ESN is not as good as the performance of the nearest neighbour algorithm and the MLP. It also suggests that in comparison to the baseline algorithms our model cannot easily be scaled to classify more different locations.

6 Discussion

Our research goals are discussed in the first subparagraph, next we give some ideas for further research in this subject and finally the practical value of our research is discussed in the last subparagraph.

6.1 Biological localization

We tried to achieve localization in a biologically plausible manner. We used different elements in our model which are biologically inspired. We used a neural network, inspired by the neurons in our brains. In comparison to feedforward neural networks our ESN is more biologically plausible. (Dominey, 1995) showed that reservoir computing shows much resemblance with some areas in our prefrontal cortex. O'keefe and Dostrovsky (1971) studied specific cells in the hippocampus of rats which become active when visiting known places. The output nodes of our ESN have the same function as these *place cells*.

6.2 Further research

There are some possible changes that can be made to the model regarding the ESN. For every location to be learned a separate reservoir can be used, leading to more specialized reservoirs. But this seems

Table 4: Summarized results, house 1

	Test conditions			
	Basic model	Gradient normalization	$\frac{1}{50}$th images	Extra places
Nearest neighbour	73,60%	99,31%	97,31%	87,33%
MLP	66,12%	99,98%	89,76%	81,92%
ESN	62,65%	99,71%	95,85%	67,84%
Lines	-	-	65,75%	-
Sift	-	-	72,50%	-

Table 5: Summarized results, house 2

	Test conditions			
	Basic model	Gradient normalization	$\frac{1}{50}$th images	Extra places
Nearest neighbour	67,55%	99,91%	98,03%	92,06%
MLP	68,72%	99,82%	66,87%	86,36%
ESN	64,54%	99,94%	58,54%	78,18%

to be infeasible for real time implementations because of the lack of scalability.

Also the pre-processing of images can be changed. The hyperbolic images could be separated in an inner and an outer ring, which are then divided in sections of 10 degrees. In that way image sections near the robot are not averaged with image sections further away of the robot, mainly walls and ceiling.

6.3 Practical value

The performance of our model was less than the performance of the nearest neighbour algorithm and the MLP, because ESN are designed to fit temporal periodic functions, not binary functions representing spatial information. The pre-processing of the images was shared by the ESN and the baseline algorithms. We have proven that it is possible to achieve accurate localization with a simple image processing technique which averages large sections of the images.

References

JL Blanco, J. González, and J.A. Fernández-Madrigal. Subjective local maps for hybrid metric-topological SLAM. *Robotics and Autonomous Systems*, 57(1):64–74, 2009.

R.A. Brooks. New approaches to robotics. *Science*, 253(5025):1227, 1991.

P.F. Dominey. Complex sensory-motor sequence learning based on recurrent state representation and reinforcement learning. *Biological Cybernetics*, 73(3):265–274, 1995.

V.V. Hafner. Learning places in newly explored environments. *Meyer, Berthoz, Floreano, Roitblat and Wilson (Eds.), SAB2000 Proceedings Supplement Book, Publication of the International Society for Adaptive Behavior*, 2000.

H. Jaeger. *Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the" echo state network" approach.* GMD-Forschungszentrum Informationstechnik, 2002.

B. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119(1):191–233, 2000.

D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

W. Maass, T. Natschlager, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002.

AC Murillo, C. Sagüés, JJ Guerrero, T. Goedeme, T. Tuytelaars, and L. Van Gool. From omnidirectional images to hierarchical localization. *Robotics and Autonomous Systems*, 55(5):372–382, 2007.

J. O'keefe and J. Dostrovsky. The hippocampus as a spatial map. Preliminary evidence from unit activity in the freely-moving rat. *Brain research*, 34(1):171, 1971.

A. Rizzi and R. Cassinis. A robot self-localization system based on omnidirectional color images. *Robotics and Autonomous Systems*, 34(1):23–38, 2001.

Z. Zivkovic, O. Booij, and B. Kröse. From images to rooms. *Robotics and Autonomous Systems*, 55 (5):411–418, 2007.

A Appendix: Experiment results

Table 6: Performances without gradient normalization, house 1

Train	Test	NN	MLP	ESN
Run 1	Run 4	72,04%	63,48%	61,68%
Run 1	Run 5	75,21%	67,54%	60,67%
Run 4	Run 1	69,92%	68,74%	66,20%
Run 4	Run 5	71,26%	60,49%	56,52%
Run 5	Run 1	79,36%	71,65%	71,52%
Run 5	Run 4	74,12%	64,83%	59,18%
Average		73,60%	66,12%	62,65%

Table 7: Performances without gradient normalization, house 2

Train	Test	NN	MLP	ESN
Run 1	Run 4	75,45%	72,37%	72,37%
Run 1	Run 5	49,80%	49,86%	49,86%
Run 4	Run 1	66,38%	66,38%	70,15%
Run 4	Run 5	64,39%	64,39%	65,50%
Run 5	Run 1	60,96%	60,96%	60,10%
Run 5	Run 4	88,30%	88,30%	79,71%
Average		67,55%	68,72%	64,54%

Table 8: Performances with gradient normalization, house 1

Train	Test	NN	MLP	ESN
Run 1	Run 4	99,03%	99,88%	99,74%
Run 1	Run 5	99,95%	99,98%	99,92%
Run 4	Run 1	100,00%	100,00%	99,95%
Run 4	Run 5	100,00%	100,00%	99,96%
Run 5	Run 1	100,00%	100,00%	99,91%
Run 5	Run 4	96,86%	100,00%	98,78%
Average		99,31%	99,98%	99,71%

Table 9: Performances with gradient normalization, house 2

Train	Test	NN	MLP	ESN
Run 1	Run 4	100,00%	100,00%	99,98%
Run 1	Run 5	99,92%	99,73%	99,95%
Run 4	Run 1	100,00%	99,92%	100,00%
Run 4	Run 5	99,60%	99,52%	99,71%
Run 5	Run 1	99,93%	99,77%	100,00%
Run 5	Run 4	100,00%	100,00%	99,96%
Average		99,91%	99,82%	99,94%

Table 10: Performances with $\frac{1}{50}$th images, house 1

Train	Test	NN	MLP	ESN
Run 1	Run 4	91,74%	82,03%	93,68%
Run 1	Run 5	99,20%	92,35%	99,31%
Run 4	Run 1	100,00%	92,80%	99,86%
Run 4	Run 5	100,00%	93,51%	99,64%
Run 5	Run 1	99,30%	93,20%	94,41%
Run 5	Run 4	93,63%	84,64%	88,21%
Average		97,31%	89,76%	95,85%

Table 11: Performances with $\frac{1}{50}$th images, house 2

Train	Test	NN	MLP	ESN
Run 1	Run 4	98,76%	75,53%	91,82%
Run 1	Run 5	95,08%	53,36%	86,99%
Run 4	Run 1	100,00%	69,47%	44,75%
Run 4	Run 5	98,26%	61,49%	35,45%
Run 5	Run 1	96,15%	66,22%	44,75%
Run 5	Run 4	100,00%	75,13%	47,48%
Average		98,03%	66,87%	58,54%

Table 12: Performances with extra places, house 1

Train	Test	NN	MLP	ESN
Run 1	Run 4	89,67%	80,44%	62,19%
Run 1	Run 5	92,70%	91,37%	74,34%
Run 4	Run 1	78,21%	73,64%	61,54%
Run 4	Run 5	87,71%	83,54%	65,15%
Run 5	Run 1	89,01%	80,65%	70,40%
Run 5	Run 4	86,67%	81,99%	73,51%
Average		87,33%	81,92%	67,84%

Table 13: Performances with extra places, house 2

Train	Test	NN	MLP	ESN
Run 1	Run 4	97,15%	89,56%	83,24%
Run 1	Run 5	92,70%	84,00%	73,58%
Run 4	Run 1	87,72%	83,31%	77,47%
Run 4	Run 5	92,70%	90,94%	78,70%
Run 5	Run 1	87,22%	78,53%	76,46%
Run 5	Run 4	94,86%	91,70%	79,62%
Average		92,06%	86,36%	78,18%