Behavior-based Perception for Autonomous Robot Navigation

Hein van der Ploeg
1007084
h.vd.ploeg@ai.rug.nl

December 2004

Supervisors
Prof. Dr. L.R.B. Schomaker (Rijks*universiteit* Groningen)
Drs. G. Kootstra (Rijks*universiteit* Groningen)

Artificial Intelligence
Rijks*universiteit* Groningen

Abstract

For navigation, many animals are known to use their visual system in combination with a process called dead reckoning, in which the animal knows its position through egomotion, for learning landmarks in an unknown environment. In robotics, previous research has been done using such an approach. The goal was to learn salient perceptual features in the environment using an unsupervised neural networ, and learn their relative locations using odometry on the robot. The resulting map showed that the information was there, but it was too approximate. The main problem was that a general sense of direction was missing and landmark representations were ambiguous.

The present study aims at using biologically inspired behaviors to yield a better encoding of the perceptual landmarks. Behaviors of this type used by animals are for example the head-scanning of rats for the detection of configurations of landmarks. These could be modeled on the robot by moving the camera in three directions, detecting three landmarks in one perceptive movement. The hypothesis is that this triple-view approach results in coupling landmarks more strictly to the environment and to each other since (1) the triple-landmark view is a less ambiguous percept than a single-landmark view, and (2) the triple-view is unique for a given direction.

Furthermore, the learning of perceptual landmarks is done on the basis of an adaptive unsupervised neural network. New items are added to the set of landmarks incrementally when needed, where the number of nodes will depend on the visual complexity of the environment.

A neural-network approach for incremental learning landmarks is proposed and tested in the first phase of the study. In the second phase, experiments were done to test the main hypothesis.

Contents

1 Introduction **4**
- 1.1 Theoretical background . . . 5
  - 1.1.1 Animal navigation . . . 5
  - 1.1.2 Previous research in robot navigation . . . 8
  - 1.1.3 Active perception . . . 9
  - 1.1.4 Unsupervised landmark learning . . . 11
- 1.2 Research goals and research question . . . 12

2 A method for incremental learning triple-landmark views **14**
- 2.1 Growing neural gas . . . 14
- 2.2 Learning perceptual input using the neural-gas algorithm . . . 18
- 2.3 Color transformation . . . 18
- 2.4 Learning the spatial layout of perceptual landmark triples . . . 20
- 2.5 Agent design . . . 22
  - 2.5.1 Proximity agent . . . 22
  - 2.5.2 Active-vision agent . . . 23
  - 2.5.3 Neural-gas agent . . . 23
  - 2.5.4 Path-planning agent . . . 23

3 Experiments and results **24**
- 3.1 Experimental setup . . . 24
- 3.2 Obtaining a dataset for training . . . 24
- 3.3 Training the Neural Gas network . . . 25
- 3.4 Recognition of landmarks . . . 29
- 3.5 Path planning . . . 34

4 Discussion **37**
- 4.1 Conclusion . . . 41
- 4.2 Future Research . . . 41

A Path-planning data of several trial runs **43**

B Usage of neural-gas program **45**

References **47**

Chapter 1

Introduction

This graduation thesis is about autonomous robot navigation. In particular, the ability for an autonomous robot to learn its environment in such a way that it is able to navigate in it in the sense of being able to plan routes from any point to any given location. In order to achieve this, a robot must have some internal representation of the world. This representation should tell the agent where it is and where other locations are with respect to that. The need for such an internal navigation system becomes clear for example in the Nasa missions where robots are sent to Mars. There are no GPS satellites orbiting Mars, at least not yet. But also closer to home, there are examples of applications where it is not desirable to rely on external systems for positioning and navigation. Robots assisting in house holding,for example, should be flexible to use in different rooms and different houses, without needing to install equipment throughout the buildings first.

As the title of this thesis indicates, this study focuses on biological strategies to obtain an internal representation for navigation. Biological research in this area have brought many interesting navigation mechanisms to light. These biological strategies have proven to be successful and it is therefore only logical to try to incorporate these in our artificial systems.

This paper will start with an overview of some of the most important mechanisms found in animals, followed by a theoretical framework for learning an environment by an autonomous robot, based on biological strategies. Then, an specific outline is given about the design of the navigation strategy and implementation on a mobile robot. At the artificial intelligence department of the university of Groningen (RuG), where this study has taken place, a robotics lab is located where the necessary experiments with the robot have been done.

1.1 Theoretical background

1.1.1 Animal navigation

Definition

Navigating animals have some ability to reach locations that are of interest to them. In its simplest form, navigation could be defined as *the process of reaching a goal location from the current location.* In this simple form, the only requirement is that the animal is able to recognize its goal when it has reached it. To be more effective, the animal must have some means of identifying where its goal location is with respect to its current location, and even better, 'knows' what path or route must be taken to get there and is able to maintain that route. The methods used for navigation differ greatly. A hierarchy of navigation techniques taken from Mallot and Franz (2000) describes the most important navigational methods known to exist in animals. It is given below.

Random search

Not particularly interesting but nonetheless a method of goal-reaching is random search. The animal will walk around in a random fashion until the goal has been reached. The only requirement for this to succeed is for the animal to be able to identify the goal when it has reached it. Insects, though equipped with more advanced navigation techniques often use this method as a backup strategy once other methods have failed. For example, when ants were displaced when walking towards their nest. The ants started walking randomly at the point where their trajectory would have reached the nest, were they not displaced (Collett & Collett, 2000).

Path integration

It is important for animals to be able to remember the location of a point of interest, for example a food source. A way of doing this is by remembering what distance and in what direction has been traveled when first finding the location with respect to another known location. The animal updates its current position relative to the point of departure by processing signals of locomotion. This intrinsic memory for travel is known as *dead reckoning* or *path integration.* The process where dead reckoning is used to navigate along a path connecting several waypoints is called *vector navigation.* The animal needs to have recorded the state of its accumulator, i.e. the dead-reckoned position, of the waypoints. Essentially, the animal performs vector-substraction in comparing the current state of its path-integrator with the recorded states, at each way-point (Collett & Collett, 2000).

Since the method is based on idiothetic clues (based on an internal reference) it is highly subjective to error. For example, in the experiment described above,

where ants were displaced when walking to their nest, the displacement caused the animals to miss their goal. The animals would continue to walk the same course, switching to random search when they did not find their goal. Furthermore, errors in path-integration will accumulate when not reset by another system. Working around these problems, the path integration system can be supported by the visual system. When an animal arrives at a location previously visited, and confirms this visually, it is able to reset the accumulator state, diminishing any errors due to path integration (Judd & Collett, 1998). When other, more advanced methods of navigation fail, path-integration often serves as a backup strategy.

Beacon navigation

Path integration is solely based on idiothetic cues, i.e., based on internal reference (Mallot & Franz, 2000). As argued, it is therefore subject to (accumulating) errors. A method of navigation which is based on external reference, or allothetic cues, is beacon navigation. Beacon navigation is also called *aiming*. When the goal location is marked by a salient feature, i.e. a visual, auditory, or other stimulus, the agent can reach the goal by maintaining a course towards that feature. In contrast to path integration, the goal can be reached from whatever point, as long as the beacon is visible (most of the time) during the trip.

Landmark-based navigation

Beacon navigation is perhaps the simplest form of landmark-based navigation, but nevertheless described separately since the other landmark-based navigation methods are more complex. A landmark is defined as a salient perceptual pattern, some object or visual pattern that stands out in its environment . There are several ways in which landmarks can aid in navigation and way-finding[1]. Experiments on bees have illustrated that visual cues of landmarks near found food sources are used to relocate the food sources: on displacement of the landmarks the bees searched for the food at the new location of the landmarks (Cartwright & Collet, 1983). Also, a route can be learned by remembering landmarks along the route, a process also called *homing* or *guidance*. This route-following behavior is observed for example in insects. Ants and bees, for example, are known to learn routes from the nest to locations of newly discovered food sources by periodically turning back to look at the food source, while walking or flying back home from it (Cartwright & Collet, 1983; Wehner *et al.* , 1996; Judd & Collett, 1998). Later, the routes are followed by comparing the current visual input to stored perceptual patterns of the landmarks.

[1] *Way-finding* involves the recognition of several places, and the ability to reach places outside the range of perception.

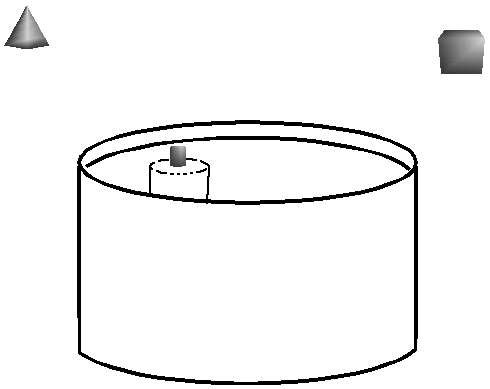


Figure 1.1: The Morris water maze. A platform is placed in the pool just below the water surface. The water in the pool is opaque, making the platform itself invisible. Distal landmarks outside the pool are visible at all times. In the training phase, the proximal landmark (the little square in the figure) is placed, which is removed at the test phase.

Also, multiple landmarks can be used to find a particular place that is not marked by a salient perceptual beacon. The angles between the landmarks as viewed from the goal-location define the location of interest. This means of encoding a location is for example seen in bees (Cartwright & Collet, 1983), and in rats and mice (Hamilton *et al.* , 2004).An experimental setup known as the *Morris water maze* has been the stage for many experiments with rats (and mice), in different navigational tasks (see figure 1.1). The setup consists of a circular pool containing a opaque (non-toxic) liquid contains an escape platform, just hidden below the water surface. Rats were learned to swim to a visible escape platform, with numerous visual *distal* landmarks outside the pool. After removal of the visual cue at the platform, the rats switched from the beacon-navigation method to a navigation method based on the configuration of the distal landmarks (Save & Poucet, 2000; Hamilton *et al.* , 2004).

Topological mapping

While insects rely mostly on path-integration, beacon-navigation and elementary landmark-based navigation for orientation and navigation, rodents seem to have a more complex knowledge of spatial layout. Tolman has shown in experiments with rats (Tolman, 1948), that even when a learned, complex route towards a food source was blocked, the rats were still able to find the food through an alternative route. The rats were given the choice of a number of alleys all leading in a different di-

rection, instead of only one as in the learning condition. Furthermore, the route towards the food which was known to the rats was blocked, so they had to choose an alternative one. Most rats now chose an alley which headed (in absolute sense) towards the food. Tolman assumed that the rats had not merely a strip-map of the maze, but, rather, a wider comprehensive map of the environment, which he later termed a *cognitive map*. These maps encode distances and angles between locations as well as sense relation (left-versus-right) in a vectorial way, see (Poucet, 1993) for a review. It is different in its representation to Cartesian grid maps, in which each known landmark is given an absolute position in a 2-dimensional coordinate frame. The use Cartesian grid maps is however not thought to exist in animals such as rats or 'lower' animals. In the formation of a cognitive map, dead reckoning is believed to play a crucial role (McNaughton *et al.* , 1996).

1.1.2 Previous research in robot navigation

There have been a number of studies in which snapshots of the environment are taken using a omni-directional camera, or a monocular camera pointed upwards toward a conical mirror, such as done by Franz *et al* (1997) . At various points snapshots are taken and compared to previously taken snapshots, analogues to the snapshot-based navigation techniques seen in insects (see section 1.1.1. The displacement of visual features on a 360° field of view obtained by a omni-directional camera is used to infer the robots location and bearing. The method proved to be successful in homing over short distances.

In recent work, Schomaker and Fehrmann have proposed a landmark-exploration model to construct an approximate 2-D map of the environment (Fehrmann, 2002; Schomaker & Fehrmann, 2003). The goal was to let an autonomous robot form a cognitive map of the environment, for which dead reckoning formed the basis.

A Kohonen self-organized map was learned from both snapshots of the environment robot and sonar readings taken by the robot. At locations where sonar readings indicated close proximity to an object, a *proximity event* was said to occur. The detected obstacle in front is possibly a perceptual salient part of the room. At a proximity event, snapshots where collected and coded in Hue and Saturation (HS). These simplified images, consisting of 30x40 pixels each having HS values, together with the sonar readings, formed input vectors for learning a 5x5 Kohonen Self-Organizing Map (Kohonen, 1990).

After training, they used this landmark map to recognize the perceptual landmarks during a live run. A matrix containing the distances between the perceptual landmarks was created by keeping track of the relative distances of the perceptual landmarks, measured by odometry, i.e., dead reckoning on the robot. Principal component analysis was performed on the distance matrix, to yield the two largest

Eigenvectors. Using these as the main axes, an approximate map of the environment was formed. On the resulting map the most prominent landmarks were placed correctly. For example, the blue and yellow goal of the robocup soccer field where the experiments took place were correctly placed at each end of the X-axis. However, the map should be scaled and rotated somewhat. In general, the map was too approximate to be a useful representation of the room.

The main problem is that when the robot is only looking in one direction, perceptual landmarks that are recognized are not coupled to other visual features in the environment. Given a distance matrix, without knowledge of relative angles, there exist multiple possible configurations of landmarks. An overall sense of direction would help couple the landmarks in the environment more strictly. Since dead reckoning forms an important basis of creating a cognitive map, angular information between landmarks is crucial. However, only with knowledge of absolute bearing the relative locations of landmarks can be measured reliably, since the angular component of a vector is dependent on the direction in which the agent or animal was facing in the first place. Such an overall sense of bearing is obtained by animals in a variety of ways, including looking at the position of the sun, a sensing of the earth's magnetic field and the polarization of light (Wehner, 1984; Shen *et al.* , 1998; Mallot & Franz, 2000).

A second problem in the approach is that some of the nodes are a very general representation of a rather large part of the room. In other words, there is an amount of ambiguity involved in the recognition of landmarks in the first place.

The study of Schomaker & Fehrmann (2003) has formed the basis for the present study. The present study also aims at creating an internal representation of an environment for which path integration and visual perception form the basis. The two components are learning perceptual salience information in the environment, i.e. learning perceptual landmarks, and learning their relative locations in the environment. The main topic is how to couple the landmarks more strictly to the environment, requiring a general sense of direction when dead reckoning is used to learn the relative landmark locations in a vectorial way. A monocular camera setup will be used for this. Although omni-directional camera would allow for much better encoding of the robot's bearing, we believe that this is biologically not very plausible.

1.1.3 Active perception

Active perception[2] denotes the use of behaviors specifically aimed at perceiving certain information about the environment or other animals/agents. It is used for exploration, in the form of scanning the surroundings and landmarks or for fixating

[2]In robotics, the use of a moving camera for tracking stationary or moving objects, for scanning or other purposes is often called *active vision*

stationary objects, or tracking moving objects (Land & Collet, 1997) such as prey. Behaviors supporting the acquisition of visual cues about a place of interest are seen for example in honeybees, which make structured flights when departing from a food source. When the bees departed from a newly discovered food-location after feeding, they performed 180 degree turns to view the entrance and its surroundings, swaying back and forth as if arriving at the food-source. This behavior, termed *turn back and look behavior* is assumed to help the bee to identify the food source for later visits (Lehrer, 1993). It allows for a better imprinting of the location, but it also helps the bee to learn the distances and angles of the food source to nearby landmarks (Cartwright & Collet, 1983).

Similarly, rats and mice have typical behaviors when exploring. When placed in a novel environment, rats and mice alternate between progression and stopping. Each time they stop, they perform scanning movements with their heads and whole-body movements, which allows them to investigate a particular location and its surroundings (Drai *et al.* , 2001; Drai & Golani, 2001).

Rats are known to be able to learn a particular location not only by its proximal cues, but also by learning its spatial relation to other more distant objects. In an experiment using the Morris water maze, rats where learned the location of a escape platform by marking it with a visible object. The pool was also surrounded by various distal landmarks. At a later time the proximal landmark was removed from the platform. When the rats entered the pool, they immediately switched to a different behavior. A short distance from the release point, they made horizontal scanning movements with their head, after which they would quickly swim to the escape point. The scanning movements are assumed to reflect attempts to sample visible stimuli, i.e., the distal landmarks (Hamilton *et al.* , 2004).

Specific behaviors such as described above are used to acquire the information needed by the animal. It is assumed that the use of active perception will also aid in the acquisition of spatial relationships of landmarks in the problem of learning an environment by an autonomous robot. Such a behavior could be making structured horizontal scanning movements with the camera upon detection of a object, sampling not only visual stimuli at close range right in front of the robot, but also more distant objects sideways. The scanning movements could be modeled by taking snapshots in three different directions: left, right and in front of the robot.

The advantages of this behavior are the following. First, as seen in the panoramic-view setups such as omni-directional camera equipped robots, the visual field as seen from one location is much bigger. With the horizontal scanning behavior each location in the field is characterized by a set of three views instead of only one. This combined view similarly results in a much bigger total view of the surroundings. This in turn makes the perceptual field at each location much more unique. At the detection of a landmark in front, the combination of three landmark views is a much

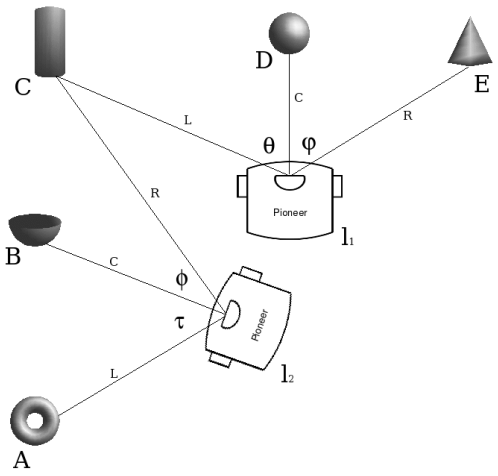


Figure 1.2: The location the robot is at, as well as the direction it is facing is captured by the a characteristic set of three landmarks seen from that pose. Also, when trying to create a map of the environment, cases where overlap takes place between the landmark sets of two stops give information about the spatial ordering of the sets of landmarks.

less ambiguous percept.

Second, the combination of three snapshots may help in determining the direction the robot is facing. A proximity event triggers a behavioral response in the robot: to look at a possible proximal landmark in front. This response is now extended by looking sideways at possible distal landmarks. The characteristic set of three views, one proximal and two distal landmark perceptions, is relatively unique for a given direction.

Finally, in the process of learning a cognitive map of an environment, cases where there is overlap between sets of landmarks give a left-to-right ordering of the landmarks involved, as figure 1.2 indicates. In the possible configurations of landmarks which follows from a table of distances between (sets of) landmarks, this ordering could serve as an extra constraint.

1.1.4 Unsupervised landmark learning

When learning the locations of landmarks in an unknown environment, the first phase is to learn to recognize perceptual salient features. We will be doing this using an unsupervised neural network. Given an input space, e.g., a collection of images taken during a run with the robot, the problem is to find a number of units which represent this input space best. This process of finding a low-dimensional subspace of an input space is called *dimensionality reduction*.

In the research of Schomaker and Fehrmann this learning of perceptual landmarks was done using an unsupervised Kohonen network. The problem with this approach is that in Kohonen networks the dimensionality of the network needs to be set in advance. This is a problem since it weakens the autonomy of the model and since the perceptual complexity of the environment is not known in advance. A very large room is likely to have more unique perceptual patterns than a smaller one. This approach is therefore both not desirable and not biologically very plausible. A more plausible method would be to have a learning scheme which allows for incrementally adding new items when needed.

1.2 Research goals and research question

The general goal of the research stream in which this study participates is to let an autonomous agent learn an internal map, either Cartesian or cognitive map-like, of an environment while exploring. This representation must be stable and robust in the sense that it needs to hold in a dynamical world where for example also other agents may be present and is usable in such dynamical conditions for navigation. Ultimately, we would like robots to be able to do this the way in which humans are capable of obtaining a representation of any environment. To reach this ultimate goal, we believe that much can be learned by studying biological systems, as well as testing them in artificial models.

As in the study of Schomaker and Fehrmann, the present study aims at creating an internal representation of an environment for which path integration and visual perception form the basis. There is a distinction between animals or autonomous agents that navigate on an internal map that they actually *have*, and those who don't, but behave *as if they had*. The difference is in their representation internally rather than the behavior outside. However, in real life it is the behavior that counts. Generally speaking, an agent may show highly intelligent behavior which is actually based on mechanisms that are far less complicated than would appear from the outside.

We already pointed out in the previous chapter that animals have very effective behaviors for learning an environment and navigation. We will be looking at the question of how such behaviors could improve perception and learning of landmarks in an unknown environment, for obtaining an internal representation of an unknown environment. The research goals are as follows.

1. For the automatic learning of perceptual landmarks, to have a learning scheme which allows for incrementally adding new items depending on the complexity of the perceptual space, as opposed to the Kohonen network, in which the dimensionality needs to be set in advance.

2. To couple the perceptual landmarks more strictly to the environment using active perceptive behavior.

The main research question of this study is:

What types of behaviors, supporting active perception, are needed to couple learned landmark representations more strictly to the environment?

Based on active exploration strategies such as the scanning movements of rats (Drai *et al.* , 2001; Hamilton *et al.* , 2004) and the turn-back-and-look behavior of bees (Lehrer, 1993) discussed in the previous chapter, we hypothesize that making structured head scanning movements, modeled by taking snapshots left and right as well as in front of the robot, will result in a more unique encoding of the landmarks. In the Morris water maze experiments, when the proximal visual stimuli had been removed, the rats switched to looking at distal landmarks. The rats inferred the correct heading by looking sideways at the constellation of the distal landmarks. Following this, the sampling of visual stimuli in three directions, modeling the scanning behavior of rats, is hypothesized to be a more informative perceptual behavior, as pointed out below.

- Three views form a more unique perceptual pattern than only one.
- The combination of the three snapshots is unique for that bearing only.
- Sense relations, i.e. a left-to-right ordering, of the landmarks are captured when overlapping of landmark sets occur.

To address the second research goal, a more plausible method of landmark learning is given in section 2.1.

An open question is how to integrate the learned landmark configurations in order to develop an internal (idiothetic) map. This issue will only be superficially addressed in this study. The current study will rather concentrate on learning an internal representation of the environment which is sufficient for the robot to navigate as if having an internal map.

Chapter 2

A method for incremental learning triple-landmark views

The two research goals reflect the two main phases in the study. First, a new method for the incremental learning of triple-landmark views is presented. This method addresses the learning of salient features in the environment using an adaptive neural network approach. The model is discussed in section 2.1. In the second part the main hypothesis is tested. To this extend, an experiment is designed, aimed at obtaining the spatial relations between sets of landmarks. These sets of landmarks consist of three perceptual landmarks which are detected by the robot at a proximity event, in each of the three directions. The method is explained in more detail in section 2.4.

2.1 Growing neural gas

As an unsupervised learning scheme, a model is proposed that has *the ability to grow when required* in the form of a neural network called *Growing Neural Gas* (Fritzke, 1995). The main idea of the method is to successively add new units to an initially small network by evaluating local statistical measures gathered during previous adaptation steps. Those parts of the input space that are covered to sparsely by the nodes in the network will lead to large errors in that part of the network. The nodes are drawn towards those sections in the input space, but also new nodes are inserted in those areas. Figure 2.1 shows how a 2- and 3-dimensional input space is covered by the nodes in the network over time. The nodes in the network spread in a gas-like manner, hence its name.

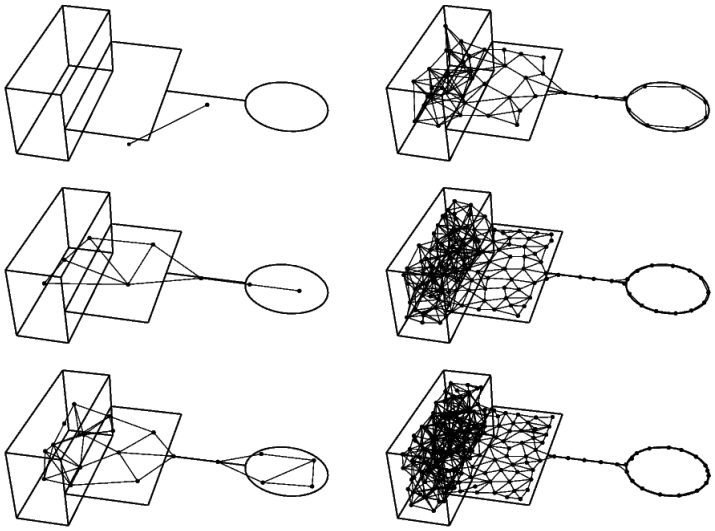


Figure 2.1: An input space is covered by the neural-gas network in a gas-like manner.

The model is described by Fritzke as follows.

Given an n-dimensional input space $\mathbf{R}^n$, the network consists of

- a set A of nodes (initially two). Each node c ϵ A has an associated *reference vector* $w \epsilon \mathbf{R}^n$. These reference vectors can be regarded as positions in the input space of the corresponding units.
- a set N of connections, or edges, between pairs of nodes. The term *topological neighbor* denotes nodes, or units which are topological neighbors in the graph (not units within a small Euclidean distance of each other in the input space).

Moreover, there is a (possible infinite) number of n-dimensional input signals ξ obeying some unknown probability density function $P(\xi)$.

The following is done repeatedly, until some stopping criterion is met:

0. Start with two units a and b at random positions w_a and w_b in $\mathbf{R}^n$.
1. Sample an input signal ξ according to $P(\xi)$.
2. Find the nearest node s_1 and the second-nearest node s_2 to ξ.
3. Increment the age of all edges emanating from s_1.

4. Add the squared distance between the input signal and the nearest node in the input space to a local state variable of node s_1:

$$\Delta error(s_1) = \|w_{s_1} - \xi\|^2 \tag{2.1}$$

$$error(s_1) = error(s_1) + \Delta error(s_1) \tag{2.2}$$

5. Move s_1 and its direct topological neighbors towards ξ, in $\mathbf{R}^n$, by fractions ϵ_b and ϵ_n, respectively, of the total distance:

$$\Delta w_{s_1} = \epsilon_b(\xi - w_{s_1}) \tag{2.3}$$

and

$$\Delta w_{s_n} = \epsilon_n(\xi - w_{s_n}) \tag{2.4}$$

for all direct neighbors n of s_1

6. If s_1 and s_2 are connected by an edge, set the age of this edge to zero. If such an edge does not exist, create it.

7. Remove edges with an age larger than a_{max}. If this results in nodes having no emanating edges, remove them as well. This is not desirable in the present study, however. This is discussed in the next section.

8. If the number of input signals generated so far is an integer multiple of a parameter λ, insert a new unit as follows.

   - Determine the unit q with the maximum accumulated error:
   - Insert a new unit r halfway between q and its topological neighbor f with the largest error variable:

$$w_r = 0.5(w_q + w_f) \tag{2.5}$$

   - Insert edges connecting the new unit r with units q and f, and remove the original edge between q and f.
   - Decrease the error variables of q and f by multiplying them with a constant α. Initialize the error variable of r with the new value of the error variable of q.

9. Decrease all error variables by multiplying them with a constant $d < 1$.

10. If a stopping criterion is not yet fulfilled go to step 1. The stopping criterion is discussed in section 2.2.

The adaptation step (5) leads to a general movement of the nearest unit and its topological neighbors towards the input signal. The accumulation of squared distances (4) helps to identify those areas in the input space where the mapping from signals to units causes much error. It is in those areas where a new node is inserted each λ^{th} iteration. This process is illustrated in figure 2.2. The uncovered part of the input space in the first part of the figure lead to a large error in that part of the network. A new node is inserted between the node with the largest accumulated error, q and its topological neighbor with the largest accumulated error, f. After a number of iterations, nodes have converged towards the uncovered areas in the input space.

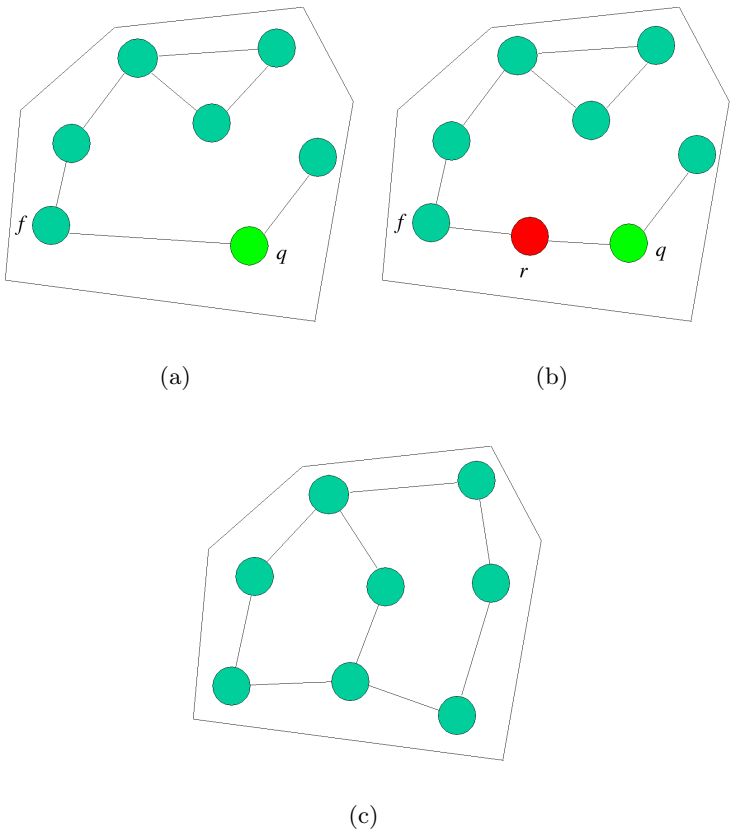


Figure 2.2: (a) The open space in the input space is not yet covered well by the network.This area will generate much error. (b) A new node r is inserted between the node with the largest accumulated error, q, and its topological neighbor with the largest accumulated error, f. (c) After number of iterations, nodes have converged towards the uncovered areas.

2.2 Learning perceptual input using the neural-gas algorithm

In the first (learning) phase, the robot will be taking snapshots and storing these to the robot's hard disk as PPM images. The PPM images are resized to 80x60 pixels. This is refined enough for the current learning scheme: the assumption is that most of the salience of perceptual features in the room is given by color-information rather than rather than texture information. Larger images would only be a bigger computational load. Taken together, the images form an input set which is used as input for the neural-gas algorithm. A single input vector of the set consists of 14400 features: 80x60 pixels each having a R, G and B value. A color transformation is done on the raw RGB values, which is discussed in section 2.3.

Although we ultimately want a robot that is capable of learning a new environment online and incrementally, the first phase, learning of perceptual input features, is done offline. Testing with the neural-gas algorithm is needed to tune parameters and doing this online is very time consuming. A separate C++ program implementing the neural-gas algorithm has been written. Its input is a perceptual-features file, containing the data gathered in a live robot run in the form of subsequent RGB values of the snapshots taken.
Since one of the criteria for the learning scheme was that no fixed network size is set in advance, some performance criterion has to be used as stopping criterion. The goal of learning is the minimization of errors. A logical choice for a stopping criterion would be the amount of error the network has with respect to the input space, or how well the input space is covered by the nodes in the network. The error each unit in the network has accumulated is a measure of distance of that unit towards the input space. The average accumulated error of all nodes will serve as a distance measure of the entire network towards to input space:

$$error_{mean} = \frac{\sum_{i=1}^{n} error_{a_i}}{n} \tag{2.6}$$

where A is the total set of nodes and n the size of A.

When *errormean* has dropped below the error threshold set in advance, learning stops. At that point, an output file is generated, which contains the nodes of the network. This file is used in recognition runs on the robot where perceptual input is compared to the learned set of nodes.

2.3 Color transformation

The robotics lab has one windowed side. Especially at sunny days, the differences in illumination are enormous because of this. The biological eye and visual system

may have very sophisticated methods of compensating for such differences in intensity. Although the camera on the robot does perform normalization on the image to account for large differences in intensity, additional normalization needs to be done. The upper row in figure 2.3 shows an typical example of the influence of the lighting conditions in the room. On a sunny day such as the day the test run was made, images are much darker when the robot faces one of the windows. And when looking in other directions, lighting from the windows illuminates some parts much more than others. The blue goal is illuminated in some parts and seems almost black to the human eye in other snapshots. The snapshots taken with the camera directed towards the windows also appear very dark.

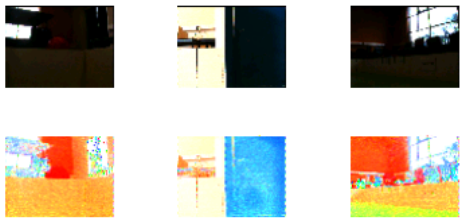


Figure 2.3: Large differences in brightness in the snapshots as a result from illumination from the window side. Below the same images after color conversion.

Without any preprocessing on the images, learning may result in an unnecessary complex network and misclassifications in recognition during a recognition run. Tests with the neural-gas network taken without any preprocessing on the images show that images are misclassified. It is assumed, that the salience in the environment will be based on color information, rather than texture information. Therefore, the proposed transformation will normalize the images by making the brightness constant.

This process is done as follows. First, the images are transformed to HSV, space. Then, they are normalized by setting the brightness (value) to maximum. Finally, they are transformed back to RGB space to make them understandable again for us. The transformation causes the images to be represented in the upper plane of the HSV cone only (see figure 2.4).

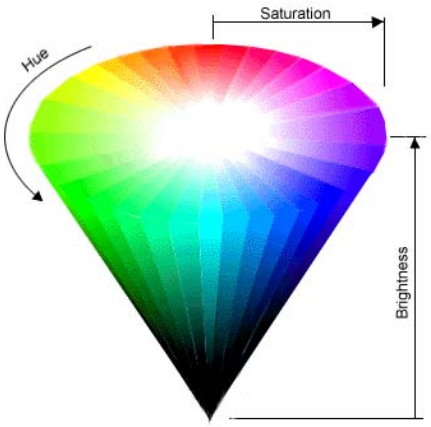


Figure 2.4: The H(ue)S(aturation)V(alue) cone. By setting the intensity to 100%, images are represented using the upper plane of the HSV cone only. The intensity or brightness of the images is made constant.

2.4 Learning the spatial layout of perceptual landmark triples

It was argued that the triple-view approach lead to a less ambiguous perception and code direction as well. The dataset contains all transitions made from one landmark triple to another. The average dead reckoned distances and angles of the transitions between two sets of landmarks could be used to generate a path to a given goal location. The resulting behavior would be a form of 'recognition triggered response navigation', in which the recognition of a set of landmarks triggers a behavioral respons: turn and move towards a desired location.
At a proximity event, the robot will stop and take snapshots in three directions, i.e., in front, left and right, at an angle of approximately 40° at each side. The images are then identified by the neural-gas agent by finding the closest nodes of a previously learned network. The combination of the perceptual-landmark IDs form a triple: the ID of the proximal landmark in front, which triggered the proximity event, and the IDs of the two distal landmarks left and right. This perceptual-landmark triple is considered a single percept.

During a run, the robot will be storing the recognized perceptual-landmark triples in a datafile. Furthermore, the distance and direction traveled by the robot relative to the last proximity event are stored as (Δx,Δy) pairs. From this, a table of transitions between landmark triples is created, illustrated below, where landmark triples are given an arbitrary name to indicate that the three perceptual landmarks combined form a single percept:

```
RW3 ... (2.3,4.5) ... PQ5
PQ5 ... (-1.5,3.0) ... GB1
GB1 ... (2.3,-1.2) ... RT2
```

From the dataset obtained by the robot in the recognition phase, a table of transitions (or a sparsely filled transition matrix) is created where each transition consists of a triple representing the start of the traject and a triple representing the end. Through vector addition, it would also be possible to generate a bearing towards a goal location directly. However, this traject could be blocked obstacles. Furthermore, the path between the two locations given by a transition is assumed to be a straight line. At a proximity event, the current percept will be used to calculate the most likely path towards the goal. Since the combination of three perceptual landmarks is hypothesized to be unique for the position as well as the direction (see section 1.1.3), the average dead reckoned direction given by the (Δx,Δy) pairs of a transition between two triples should be approximately correct. The main idea is to calculate a path towards a given goal location and determine the direction towards the first subgoal of the route, illustrated by the following.

- A transition is defined as a straight path between two landmark triples, start and end.
- Since the center landmark is defined as the proximal landmark, let the goal location be defined as any landmark triple which has the goal ID as center percept. Any such triple is a goal triple.
- A path is defined as a number of transitions in which the first transition starts at the current triple, i.e., the robots current percept, and the end triple is a goal triple.

When following a route, the robot is said to on track when the detected center landmark at a proximity event is the same as the expected center landmark at a given stop.

At a proximity event, determine the next action to be taken as follows:

1. Find the first transition in the table of transitions for which the end triple is a goal triple.
2. Find in the table all paths starting at the current triple leading to the goal triple.
3. When no path is found, start random search behavior, i.e. obstacle avoidance. At the next proximity event, proceed with step 2.

4. When one or more paths are found, calculate for every path a measure of being successful. Let N be the number of times a single transition, T_j, of the route has been driven before. E is the number of times the robot has driven any transition starting at the start triple of T_j. l is the length of the path, i.e., the number of transitions in the path. The probability p of the path being successful is given by

$$p_{path} = \sum_{i=1}^{l} \frac{N_i}{E_i} \tag{2.7}$$

5. Chose the path for which p_{path} is largest. This is the path that is most likely to be successful.

6. Find the dead reckoned distance and angle of the first transition of that path, relative to the current position.

7. Calculate the bearing towards that location, turn accordingly and start driving in that direction.

8. At the next proximity event, check whether you are still on track by checking whether the center-ID of the current triple is the center-ID of any triple along the path. If so, go to step 6. If not, go to step 1.

2.5 Agent design

2.5.1 Proximity agent

The proximity agent is responsible for the default driving behavior of the robot. Essentially, it is a basic obstacle avoidance agent, but with an added behavior. In presence of an object the robot now needs to take three snapshots, for which the robot needs to stop. The proximity-agent is designed as follows.

- The default behavior is to drive forward.
- When sonars detect an obstacle in front (in a range of approximately 60°) within one meter, a proximity event is said to occur. The robot then stops the robot for the purpose of scanning the surroundings. A stop is not made when less then one meter has been driven since the last stop.
- Obstacle avoidance behavior steers away from obstacles, when less than one meter has been driven since the last stop.

2.5.2 Active-vision agent

The active-vision agent implements the scanning behavior. When a proximity event occurs, the robot is stopped by the proximity agent to give the active-vision agent time to look around. It will take a snapshot in three different directions, in front and at an angle of 40° at each side. These snapshots are stored on the robot's hard disk as PPM images.

For the purpose of the following experiments, there are three different modes implemented: learning, recognizing and planning. The learning mode is used to create a dataset for the learning phase of the research. The snapshots are stored on the robot's hard disk to form an input set for the neural network. In recognition mode, the robot will be matching its visual input with nodes in the learned neural-gas network. At every stop, the perceptual landmarks that are detected as well as the robot's position relative to the previous stop (Δx,Δy) are stored in a datafile. The perceptual landmarks are stored as IDs as given by the neural-gas agent. This information is used by the path-planning agent for navigation. The path-planning agent is described in section 2.5.4.

2.5.3 Neural-gas agent

The neural-gas agent is basically a stripped-down online version of the neural-gas program used offline learning as described in 2.2, which is used to recognize previously learned landmarks. The output file generated in the first learning phase is read when the agent comes online. At each stop, the active-vision agent passes the image to the neural-gas Kohonen self-organized map was learned from both snapshots of the environment robot and sonar readings taken by the robot. At locations where sonar readings indicated close proximity to an object, a *proximity event* was said to occur. The detected obstacle in front is possibly a perceptual salient part of the room. At a proximity event, snapshots where collected and coded in Hue and Saturation (HS). These simplified images, consisting of 30x40 pixels each having HS values, together with the sonar readings, formed input vectors for learning a 5x5 Kohonen Self-Organizing Map (Kohonen, 1990).
agent, which calculates the nearest neighbor of the image in the network using the distance measure (eq. 2.1).

2.5.4 Path-planning agent

The Pathplanning agent is used in the last experiment (section 3.5) to generate the best path from the current position to the goal location. Given a dataset of transitions taken during a previous run and the IDs of the landmarks recognized at each direction, the path planner calculates the relative X and Y position of the first subgoal of the most likely path, based on previous travels between the same sets of landmarks.

Chapter 3

Experiments and results

3.1 Experimental setup

The robot used in this research is a Pioneer II DX type mobile robot, which is basically a Pentium class computer at 266 Mhz running linux. Its perceptual system consists of an array of 16 sonars, and a monocular CCD camera. The sonars are capable of detecting objects in a range of about 10 cm to about 5 meters or more. The camera is mounted on top of the robot. It has an effective resolution of 752 H x 585 V pixels and is able to move both horizontally and vertically at a range of about 200° H (pan) and about 50° V (tilt). For path integration purposes, the Pioneer robot measures each wheel's traveled distance independently.

The robotics lab of the University of Groningen consists of a robocup soccer field, situated in a room having windows at one side and a wall at the opposite side. Part of the walls is colored orange. The robocup soccer field consists of a green floor, white boardings of about 40 cm high, and two goals of about 1 meter high colored blue and yellow respectively. Part of the walls in the room are colored orange. This environment for the research and no extra artificial landmarks will be added to the room. Since the boarding of the soccer field is quite high, the camera is tilted 5° upwards to enable the robot to see more of the surroundings instead of white boarding for most of camera view.

3.2 Obtaining a dataset for training

Before training of the neural network can take place a set of snapshots is needed as a dataset. For this a run with the robot is made in which snapshots are taken in all three directions at each proximity event. This run was made on two different days with slightly different weather conditions, but with both the curtains open and closed. At each day, the robot drove for about four hours resulting in a dataset of approximately 2100 snapshots. This means that with a network of 21 nodes, each

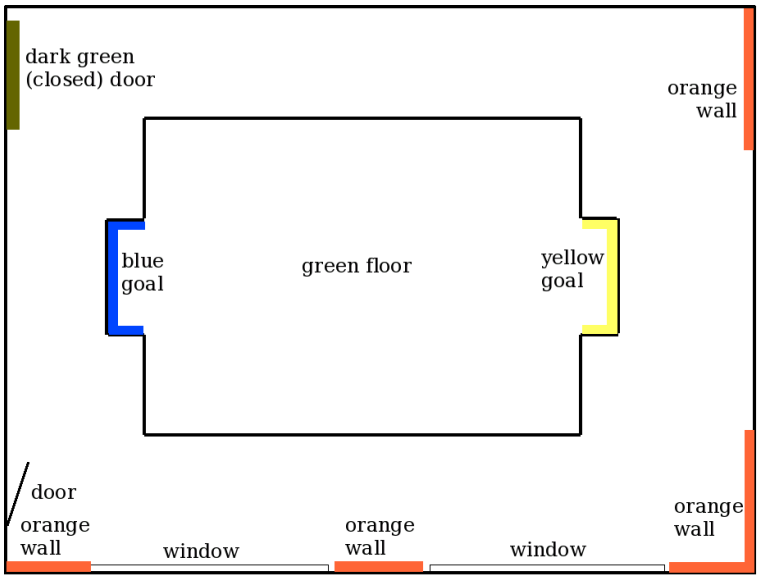


Figure 3.1: The robotics lab. Part of the walls is colored orange and at one side the room has windows. Around the soccer field (in the middle) there are computer terminals and red chairs.

node will have had an average of 100 examples.

3.3 Training the Neural Gas network

In order to get reasonable results after learning we performed some testing on the parameters of the Neural Gas network. A list of the parameters used in the network is given below. A file containing these parameters is loaded by the neural gas program at execution.

- **MAX_NODES** (2 - ∞) Defines the maximum number of nodes.
- **MAX_ERR** (0 - ∞) The maximum error. Learning stops if the mean error of all nodes drops below this value.
- **LAMBDA** (1 - ∞) Each λ^{th} iteration, the network inserts a new node in the network in the area containing the largest error with respect to the input (step 8 of the algorithm).

- **ALPHA** (0 - 1) The parameter α determines the error decay for insertion. When, at step 8 the network inserts a new node in the network, the error of the units with the largest- and second-largest error q and f respectively, is lowered by multiplication with α.

- **AGE_MAX** (1 - ∞) This parameter defines the maximum age of an edge before it is removed.

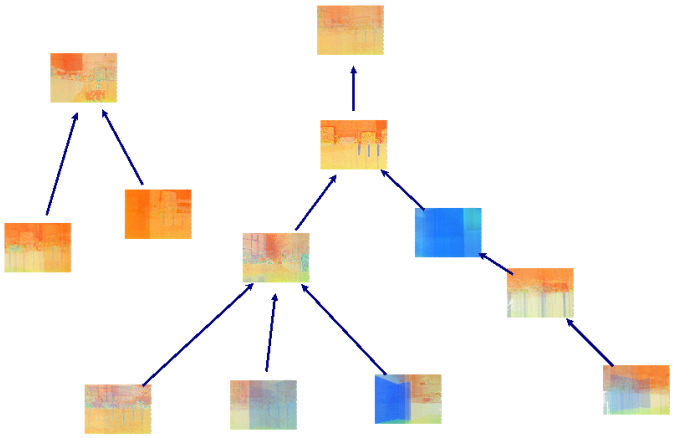
- **EPSILON_B** (0 - 1) The winner node for a particular input vector ξ is moved towards ξ with fraction ϵ_b. that in

- **EPSILON_N** (0 - 1) All topological neighbors of the winning unit are moved towards the input vector with fraction ϵ_n.

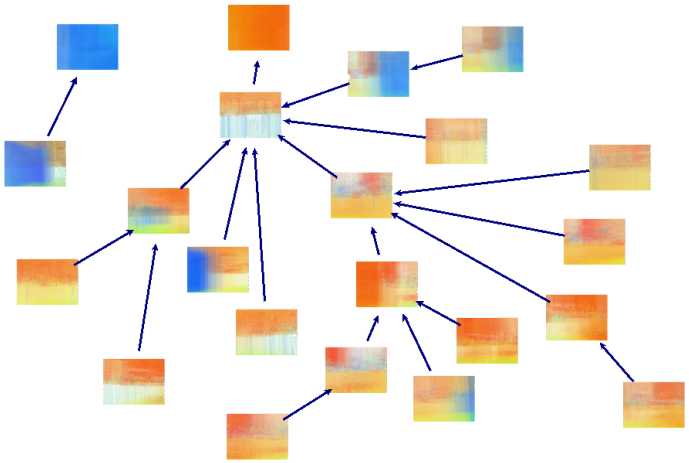
- **ERROR_DECAY** (0 - 1) In step 9 of the algorithm, the error variables of all units are lowered by multiplying them with error decay β. This will keep the errors of growing out of proportion and will increase the influence of the error of younger nodes.

Since λ determines the number of cycles after which a new node is inserted, when λ is set to a high value learning may take very long. When set to a low value, nodes are moved around a lot without moving towards a certain region in the input space.

When the age of an edge exceeds threshold AGE_MAX, the edge is removed, possibly deleting a node in the process. Thus, when a certain unit or its neighbor hasn't been winner node for some time it has become obsolete. However, in the case of landmark learning it is not desirable that a potential landmark is deleted from the network because input features supporting it haven't been presented to the network in a long time. The parameter AGE_MAX must be set so that no, or few nodes are deleted in the learning process, but restrain the amount of edges to keep the network from becoming globally interconnected, in which case all landmarks would be affected at each iteration.



(a) Nodes in the network after learning with $\epsilon_b = 0.4$, $\alpha = 0.4$ and $\beta = 0.90$. With these settings the original images are shown too clearly: the network is not generalized enough. Also, multiple parts of the room are shown in single nodes, which makes the locations represented by those nodes ambiguous.



(b) Nodes in the network after training with learning parameters $\epsilon_b = 0.08$, $\epsilon_n = 0.007$, insertion parameter $\lambda = 130$ and a maximum error of 5000 on a dataset of 2100 features.

Figure 3.2: Two networks after training. The arrows indicate each node's parent: the node with the largest accumulated error at the time of insertion of the child node.

ϵ_b and ϵ_n are the parameters for the learning rate. When ϵ_b is set to a high value, the winner node at each iteration moves towards the present input vector faster. The topological neighbors of the winning node are also moved towards the input vector, but with the much smaller ϵ_n. Generally speaking, the network stops learning sooner when ϵ_b is set to a high value. Setting the values too low will lead to slow and ineffective learning. Setting them too high will result nodes moving towards the input vectors too quickly. With smaller values of the error-after-insertion parameter α, and error decay β, learning stops too early, too. The effect of this is that the resulting network will be unstable and generalized insufficiently.

In some experiments, these settings clearly made the stopping criterion being met too soon, when only a small part of the input set had been presented to the net. Figure 3.3(a) shows that with these kind of settings, the nodes in the network after learning start showing original images from the dataset too clearly. These nodes are then representing a rather small set of the input space, while other parts of the input space are not covered enough. This amount of nodes seems too little to map onto the entire input space well. A number of nodes in the figure are showing different parts of the room, which will make perception by the robot ambiguous.

To ensure that each node would have had enough examples from the training set, a criterion was that at least the entire dataset should have been presented to the network after training. The learning rate must therefore not be set to high, and the insertion parameter not to low. With the dataset of approximately 2100 features a network of 22 nodes was obtained with the parameters for learning $\epsilon_b = 0.08$ and $\epsilon_n = 0.007$ and the insertion parameter $\lambda = 130$. The accumulated errors of the individual nodes are given in table 5.1.

Node ID	$error_{rms}$	Node ID	$error_{rms}$
0	3118	11	5756
1	3894	12	1719
2	11293	13	10364
3	6611	14	5252
4	2754	15	352
5	491	16	2865
6	4890	17	6376
7	5501	18	5119
8	7535	19	1250
9	9024	20	42
10	4595	21	1693
Mean 4568			
St. Dev. 3083			

Table 3.1: The accumulated rms error per individual node of the network after training.

This amount of nodes seems to capture the most important visual characteristics of the room, without creating too much overlap between nodes (see figure 3.3(b)). The blue goal emerges most clearly, since this is the only large blue object in the room. Most other nodes are colored yellow since part of the walls of the robot lab is colored orange and the artificial-light (TL-lighting) emits yellowish light also. The yellow goal is therefore somewhat more hidden but is nevertheless there in a number nodes. There seems to be one or two general nodes for the non-window side of the room, but they are difficult to locate in the real room.

In the recognition experiment described in the next section, it is tested how stable the trained networks are over different days, with different lighting conditions. Also, it is tested what the effect of different network sizes yielded by different parameter settings is on the error in recognition (see figure 3.4).

3.4 Recognition of landmarks

In this phase the robot is taking snapshots in the three directions at proximity events, but now uses the learned set of landmarks and the neural-gas algorithm to compare the taken images to the learned landmarks. At each proximity event the identity of the landmark recognized in each direction is written in a log file, along with the dead reckoned distance between the current stop and the previous stop. To test whether the proposed model is able to recognize landmarks well enough, a test

run was made with the robot. During this run, the robot's behavior is similar to the learning phase except that it now not only stores snapshots, but also classifies each taken snapshot as one of the learned landmarks, following equation 2.1. This, together with the dead reckoned distance from the last stop to the current position is stored in a data file. The robot was run for a period of 8 hours on two different days, which resulted in a datafile containing the snapshots and distances traveled of 583 stops.

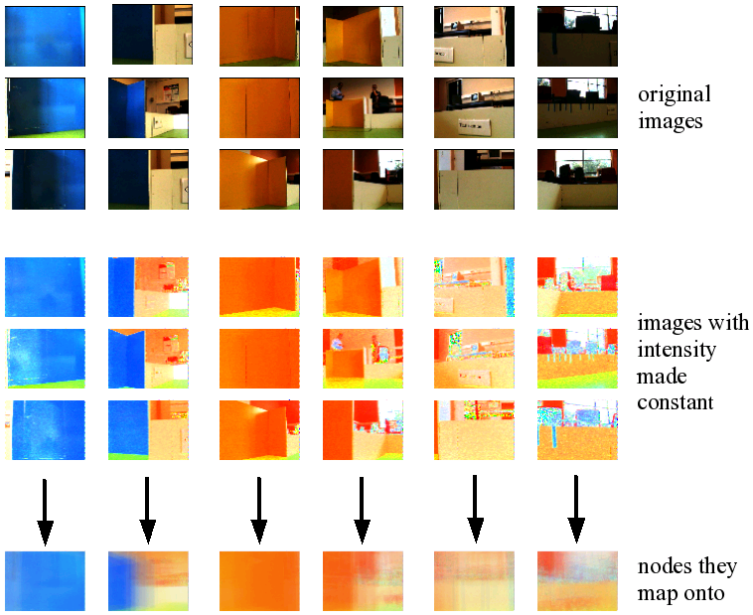


Figure 3.3: Some examples of mapping of snapshots onto nodes during a typical recognition run with the robot.

Judging the performance of the model in a recognition run is a relatively subjective task. The nodes as learned by the network are not all easily identifiable as exact locations in the real room. For this reason, the robot sometimes makes a choice for a node given an snapshot which the human observer would not have made, based on prior knowledge about the room. The model just calculates the Euclidean distances between nodes and snapshots, and is therefore always 'right'. To give an impression of the recognition process during a typical run, figure 3.4 shows a number of examples snapshots mapping onto nodes. Observing the recognition during the runs with the robot, the algorithm makes the correct choices in almost all cases where there is little ambiguity in the nodes themselves such as the blue goal. At the darker side of the room there is somewhat more ambiguity. The model sometimes chooses

the same (ambiguous) node at locations relatively far from each other (but at the correct side of the room). However, the landmarks recognized in the other directions together still seem to make sure that a set of landmarks represents the view from that particular location.

A more objective measure of performance is the error made by the network, i.e., the Euclidean distance between snapshots and nodes (rmse). To this extend, datasets were created of different runs, each containing approximately 600 to 700 snapshots. Offline, the neural-gas network was used to calculate the average error of the entire runs with different network sizes. This gives an impression of stability of the network over time, where lighting conditions may change. Furthermore, it gives an impression of the performance of networks of different size.

In table 3.4 and figure 3.4 the average distances of different networks are given. The lighting conditions on the subsequent days differed from sunny to clouded, but without rain and before sunset (figure 3.4). It is striking to see that the average distances of the networks in each run differ only by a small margin. As would be expected, the errors are smaller in the case of the larger network of 36 nodes. However, the differences are only small, which indicates that the smaller network maps onto the input space well enough. A network of size 12 gives a much greater error in mapping. As for stability, the model seems to perform relatively equal on all days, with a small increase in error on one, clouded, day.

jul 29	Sunny
jul 30	Partly clouded
aug 02 (noon)	Sunny
aug 02 (afternoon)	Partly clouded
aug 04	Clouded
sep 27	Sunny

Table 3.2: Lighting conditions on the test days.

	Network size 12		Network size 22		Network size 36	
Date	Mean Dist.	St. Dev.	Mean Dist.	St. Dev.	Mean Dist.	St. Dev.
jul 29	6724	1503	5867	1582	5637	1501
jul 30	6644	1322	5843	1370	5609	1293
aug 02	6693	1448	5884	1568	5718	1484
aug 02	6909	1440	6153	1448	5973	1389
aug 04	6903	1439	6103	1444	5922	1381
sep 27	6587	1418	5830	1477	5643	1388

Table 3.3: Average rmse between snapshots and nodes with different sets of snapshots, taken on different days.

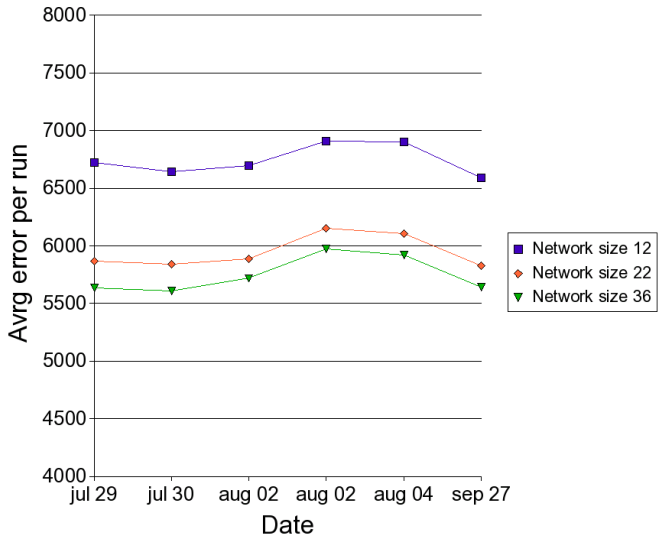


Figure 3.4: Average distances between snapshots and nodes with different datasets of snapshots, taken on different days.

The previous section described how the visual space becomes unnecessarily complex when the raw RGB values of the snapshots are taken as input for learning. Learning without preprocessing on the images has been done to test the effect of the image normalization. The network shown in figure 3.5 resulted from learning with the same parameters as used in the previous section and on the same set of snapshots, only without color conversion on the images. The network has become one node larger than without preprocessing where a number of nodes represent almost

the same part of the room. Figure 3.6 indicates the difference in the rms error.

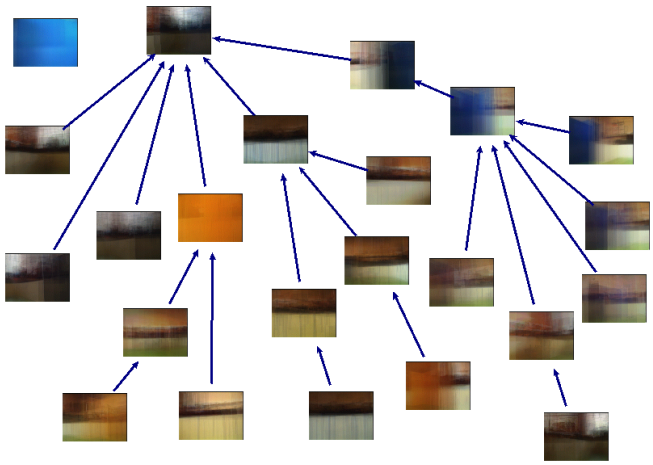


Figure 3.5: The network after learning without preprocessing on the images. This network was obtained by training with the same parameter settings as those which yielded the network of size 22 (figure 3.3(b)).

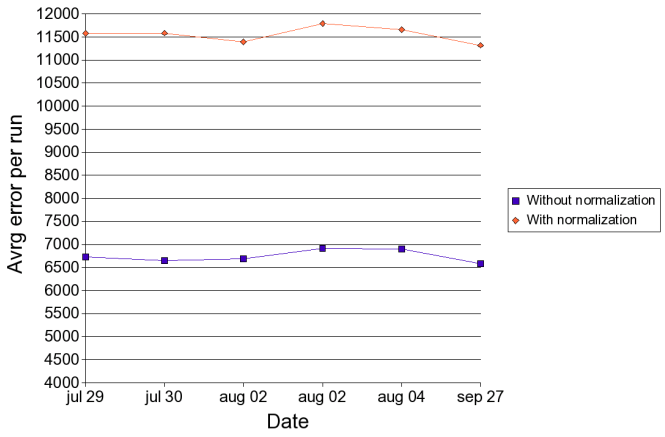


Figure 3.6: Difference in average rmse in recognition runs between networks with and without preprocessing on the images. The lower line indicates the average rmse of the network of size 22, where normalization was done on the images. The upper line indicates the average rmse of the network obtained by training with the same parameter settings, but without normalization.

3.5 Path planning

The following experiment has been done to test how well the obtained representation of the room can be used in reaching a given goal landmark. The method explained in section 2.4 was used in various runs with the robot, with different goal locations and different starting positions. At each run, the number of transitions needed to reach the goal was recorded. The average number of transitions needed to reach a goal when driving randomly, N_{random}, was taken as baseline:

$$N_{random} = \frac{N_{trans}}{N_{goal}} \tag{3.1}$$

where N_{trans} is the total number of transitions of all runs and N_{goal} the number of transitions in which the goal ID was encountered at the end of the transition.

Figure 3.7 and table 3.5 show the resulting data.

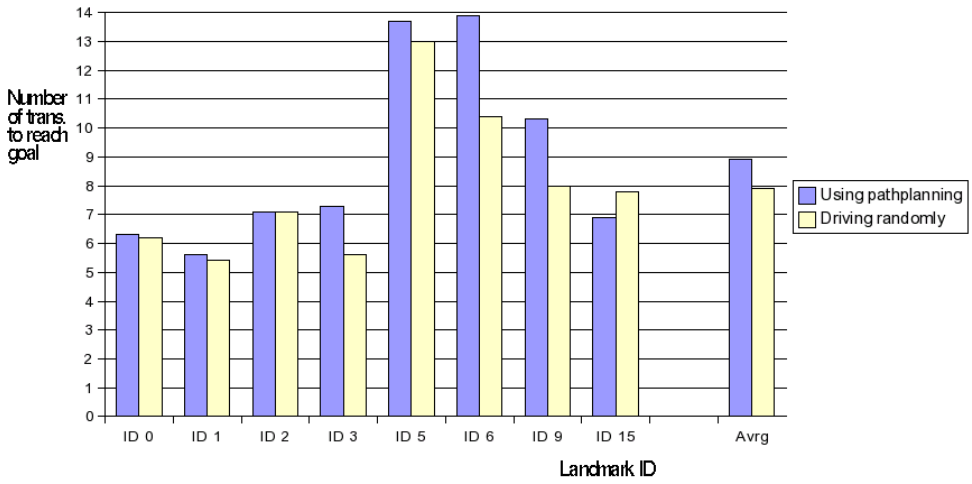


Figure 3.7: Average number of transitions to reach a given goal landmark when using the path-planning algorithm and when driving randomly. Error bars are not given, since the spread is rather large. See table 3.5 for this information.

Node id	Min	Max	Mean	St dev
0	2	17	6.3	3.6
1	2	12	5.6	2.4
2	2	17	7.1	3.6
3	2	19	7.3	5.0
5	3	47	13.7	13.3
6	3	29	13.9	7.6
9	3	36	10.3	9.4
15	3	19	6.9	4.3

Table 3.4: Average number of transitions needed to reach goal using path planning. The minimum values, maximum values and the standard deviations of the data are also given.

The average number of transitions needed to reach the entered goals, compared to the performance at random drive are shown in figure 3.7. It is clearly shown that the performance of the path planner is less than at random drive. Also, as indicated in table 3.5, the spread is large. The smallest number of transitions needed for each goal location is small, i.e. 2 or 3. However, there number increases to as much as 47 in one trial. The number of transitions needed by the model is almost system-

atically higher than random search, which is rather strange. It could indicate that the information is there, but is used in the wrong way. We will further discuss these results in the next section.

Chapter 4

Discussion

Starting with the problem of unsupervised learning of perceptual landmarks, an adaptive learning scheme was proposed. This model has been based on a neural network structure called *Growing Neural Gas* (Fritzke, 1995). As the Kohonen network, the GNG network is a method of reducing the dimensionality of an input space, in our case a collection of snapshots of a room taken by the robot, to a set of nodes which map onto the input space best. In our implementation, the model successively adds new items until the overall error (expressed by the squared distance of all units to the input features) is considered small enough.

After experimentation on tuning of the parameters, the model was given an input set of approximately 2100 images After learning the model had created a network of 22 nodes, which covered the input space well, judging subjectively. In runs with the robot, testing the performance in recognition, network sizes of 12 and 36 were also taking into account. The average error (rmse) between snapshots and learned perceptual landmarks in different runs with the robot indicated that the network size of 12 resulted in a much larger error and the network of 36 nodes produced only a slightly smaller error. Furthermore, the performance of the network with different lighting conditions was relatively constant, with a slight increased in error in recognition when it was (partly) clouded. The results indicate that the representation is stable.

The second stage of the study was aimed at capturing the relations between the landmarks in the environment. During the recognition run with the robot, a datafile with information about the detected landmarks at each stop, and the dead reckoned distances between subsequent stops was created. From this a distance matrix was created containing the average x and y position of perceptual-landmark triples relative to each other. At a proximity event, given the percept of three landmarks, the most likely path to a given goal location was calculated. The performance of the algorithm was compared to random-drive behavior, i.e. the number of transitions driving randomly.

The experiment had a rather surprising result: the number of transitions needed to reach the goal was systematically higher than when using path planning. It seems that the information is there, but is used in the wrong way. Observing the robot's behavior in the experiment, two important things were observed.

First, looking at the perceptions of the robot during the test, it was often seen that the direction the robot traveled in was approximately correct. However, especially at larger distances a small error in the bearing results in a much larger error at the end of the traject, leading to a 'miss'. While observing the robots behavior, it could be seen that there was a number of cases where the expected proximal landmark was indeed detected, but not at the center location. Examples of this can be seen in appendix A, where a table is displayed in which the detected and the expected landmark IDs at each stop in a number of trials is given. Visualization of a run in figure 4.1 shows an example. Here, the first subgoal generated is not reached, and the robot is too far off to the left. In stops 3 and 4 however, the robot's perceptual fields are not that different from what was expected. Also, the situation sketched in figure 4.2 indicates the situation where a small displacement causes the model to detect a different landmark than expected. Perceptually, the two nodes differ only by a small portion, and the human observer might even consider the nodes as different representations of the same landmark.

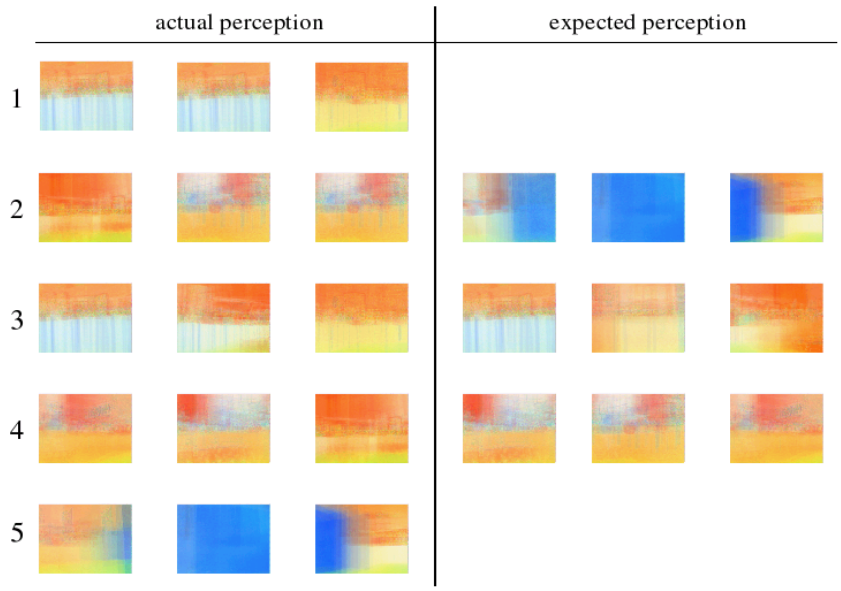


Figure 4.1: A given run to find goal ID 0. In first column the perceived landmarks at each stop, in the second column the landmarks that the robot expected to see at that stop.

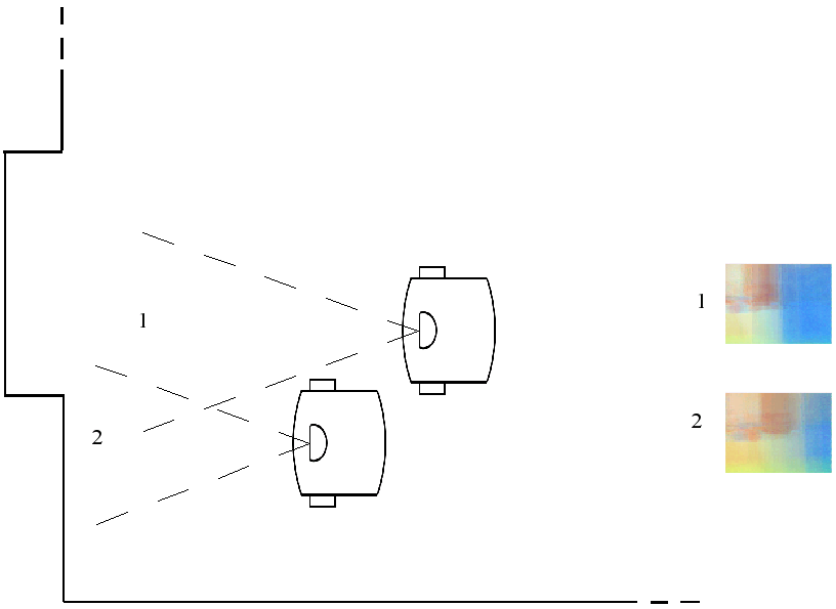


Figure 4.2: A small displacement causes the model to detect a different landmark than intended. Perceptually, the two nodes differ only by a small portion. However, cross correlation on (horizontal strips of) the images would yield information for corrections in direction.

Second, while observing the behavior of the robot during path planning it was often seen that the robot took of in approximately the correct direction but changed course along the way. The Proximity agent makes sure that a certain distance is traveled before making another stop, to keep the robot from stopping endlessly at one location. This behavior is correct most of the time, but made the robot change direction in a number of cases resulting in missing the intended goal.

Third, there are too many cases where no path had been found at all. These cases arise when the detected set of landmarks has not been seen before, if any path generated is longer than the maximum allowed size of a path or no path exists at all in the dataset. The chance of finding a good path would be greater if the amount of data collected during the recognition run was larger. A rule of thumb would be to have at least 10 examples for every transition. However, the number of possible transitions is huge. A network of 22 nodes would lead to a distance matrix of size 22^3 by 22^3, which is 10648x10648. Even though the matrix is sparsely filled, it is still very large. The time needed to gather the amount of data required to have 10 examples for every transition is very long, which is not desirable, biologically speaking.

Generally speaking, in the current approach the robot chooses a direction based on a given percept but after that relies entirely on path integration to reach its goal. Turning towards some landmark you wish to reach and then starting to walk towards it with your eyes closed seems not a very wise strategy. As indicated in section 1.1, when goal locations are marked by a salient beacon, insects aim at this beacon to reach the location. Also, when following a route landmarks are used stick to a track and correct disturbances and errors in path integration (Collett, 1996). A form of beacon navigation during the driving phase could give the robot the necessary information to make small adjustments in its direction. Comparing the image of the goal location with the current visual input at different directions would indicate the direction most likely to be successful.

4.1 Conclusion

The hypothesis of this study was that a modeled form of the scanning behavior of rats would lead to a better encoding of the perceptual landmarks in the environment, since the percept of three perceptual landmarks is less ambiguous and since this percept is unique for one direction. The last experiment, where path planning was used to reach a given goal location was most important in testing this hypothesis. The results of this experiment were that using path planning, the robot reached goal locations less often than when it was driving randomly. However, observations pointed out that given a perceived landmark triplet, the expected direction of the next goal calculated by the model was approximately correct. The goal was mainly missed because of smaller errors in the direction, and changes in direction made by the obstacle-avoidance agent, when they were not desired. The main problem would be that during the driving phases between subsequent stops, the robot is blindly relying on path integration and the obtained information, rather that visually confirming it is on the right track.

4.2 Future Research

Future research could be directed at implementing a form of beacon navigation to backup the current strategies. In the driving phase between subsequent proximity events, perceptions in a number of directions could be used to make small corrections in direction. It is known that insects, i.e. bees and ants, match the their visual percept to stored percepts of landmarks for guidance and aiming (Cartwright & Collet, 1983; Judd & Collett, 1998). In doing so, they process not the raw retinal images, but rather a processed version that emphasizes edges (Collett, 1996). If the image is displaced on the retina with respect to the learned pattern, the animal changes direction so that it matches again.

Following this, comparing (parts of the) snapshots taken during the driving phase

to the learned perceptual landmarks, could be done using cross-correlation on small horizontal parts of the images. The horizontal displacement of images in comparison to the goal image indicates the amount of error in direction.

As a second point, this study did not have the intention to let an autonomous robot create a *cognitive* or *Cartesian* map of the environment, but rather at a behavior in which it would seem that it had. A solid map-like representation of the environment would however allow for more efficient navigation and path planning, especially in highly dynamical environments, where routes may have become blocked and alternative routes must be taken. As indicated in section 1.1.3, the combination of three landmark percepts at a proximity event not only couples the three perceptual landmarks in the triple, but also those of other triples when triples overlap (indicated in 1.2). The left-to-right relations between the perceptual landmarks are an extra constraint in placing the perceptual landmarks on the map.

Appendix A

Path-planning data of several trial runs

[H]

transition nr	actual set of IDs left-center-right	expected set of IDs left-center-right
1	16 - 5 - 6	—
2	8 - 13 - 2	10 - 13 - 2
3	5 - 0 - 7	20 - 0 - 7
1	2 - 14 - 19	—
2	1 - 1 - 15	1 - 1 - 18
3	3 - 3 - 16	—
4	14 - 4 - 1	—
5	6 - 10 - 13	10 - 13 - 2
6	19 - 18 - 15	18 - 15 - 16
7	2 - 14 - 19	—
8	10 - 1 - 9	1 - 1 - 18
9	3 - 15 - 18	3 - 15 - 18
10	0 - 0 - 6	5 - 0 - 6
1	15 15 - 16	—
2	13 - 2 - 14	0 - 0 - 6
3	19 - 3 - 3	3 - 12 - 0
4	13 - 14 - 11	2 - 11 - 4
5	9 - 3 - 16	—
6	18 - 3 - 3	—
7	0 - 0 - 6	20 0 6
1	17 - 13 - 14	—
2	1 - 1 - 9	—
3	3 - 11 - 5	16 - 16 - 12
4	10 - 13 - 14	—
5	2 - 14 - 19	—
6	1 - 9 - 15	1 - 1 - 18
7	16 - 3 - 12	3 - 21 - 10
8	5 - 0 - 6	—
1	2 - 2 - 10	—
2	19 - 3 - 3	5 - 0 - 6
3	2 - 14 - 10	2- 11 - 4
4	16 - 15 - 19	15 - 3 - 16
5	12 - 0 - 6	—

Table A.1: Data from various runs in which the ID 0 was the goal-ID, the blue goal. For each trial, the recognized landmark IDs at the stops are indicated in the second columnm, and the landmark IDs that the robot expected to see at that point are given in the third column. Dashes indicate that the robot did not find a path at the previous stop, thus no landmarks are expected.

Appendix B

Usage of neural-gas program

Growing Neural Gas Network Taken from 'A Growing Neural Gas Network Learns Topologies', Bernd Frizke, in *Advances in Neural information Processing Systems* 7, 1995

To learn a set of feature vectors, a input file is needed containing the following: On the first line the length of the feature vectors and the minimum- and maximum value of a feature, seperated by a space. Each following line should be a feature vector of length provided above, in which features are seperated by spaces. As output, the network by default writes a file 'output' to the working directory. This file has the same makeup as the input-file.

Network parameters are stored in "gng.ini" which is read by the program at execution. The network uses two stopping criteria: A maximum number of nodes and a maximum error. Both should be fairly obvious. Setting the maximum number of nodes to a high value makes the network use the other as stopping criterion, while setting the error 0 makes the net grow until *max_nodes* is reached.

A topology file is written after learning is complete. This file is made up of two parts, one containing the set of edges between nodes, and the other each node's parent. Each λ-th iteration, a node is spawned between the node with the largest accumulated error and its topological neighbour with the largest accumulated error. This *largest_error_node* is the spawned node's parent.

Example topology file:

```
0 2       ** The set of edges between nodes, N.
0 3       ** (node's and their neighbours)
1 2
1 4
2 3
3 4
####
2 1       ** Each node's parent. The first and second
3 1       ** node (0 and 1) do not have parents as they
4 3       ** were spawned at initalization.
```

References

Cartwright, B.A., & Collet, T.S. 1983. Landmark Learning in Bees. *Journal of Comp. Physiology*, **151**, 521–543.

Collett, M., & Collett, T. S. 2000. How do insects use path integration for their navigation? *Biological Cybernetics*, 245–259.

Collett, T. S. 1996. Insect navigation *en route* to the goal: multiple strategies for the use of landmarks. *Journal of Experimental Biology*.

Drai, D., Kafkafi, N., Benjamini, Y., Elmer, G., & Golani, I. 2001. Rats and mice share common ethologically relevant parameters of exploratory behavior. *Behavioural Brain Research*, **125**, 133–140.

Drai, Dan, & Golani, Ilan. 2001. SEE: a tool for the visualization and analysis of rodent exploratory behavior. *Neuroscience & Biobehavioral Reviews*, **25**, 409–426.

Fehrmann, R. 2002. Cognitive navigation modeling in robots. *MSc Thesis, University of Groningen, A.I. Department*.

Franz, Matthias O., Scholköpf, Bernhard, Mallot, Hanspeter A., & Bülthoff, Heinrich H. 1997. Where did I take that snapshot? Scene-based homing by image matching. *Biological Cybernetics*, **79**, 191–202.

Fritzke, Bernd. 1995. A Growing Neural Gas Network Learns Topologies. *Advances in Neural Information Processing Systems*, **7**.

Hamilton, Derek A., Rosenfelt, Cory S., & Whishaw, Ian Q. 2004. Sequential control of navigation by locale and taxon cues in the Morris water task. *Behavioural Brain Research*, **154**, 385–397.

Judd, S. P. D., & Collett, T. S. 1998. Multiple stored views and landmark guidance in ants. *Nature*.

Kohonen, T. 1990. The Self-Organizing Map. *In: Proceedings of the IEEE*.

Land, M.F., & Collet, T.S. 1997. A survey of actvie vision in invertebrates. *Pages 19–36 of:* Srinivasan, M. V., & Venkatesh, S. (eds), *From living eyes to seeying machines*. Oxford, New York, Tokyo: Oxford University Press.

Lehrer, M. 1993. Why do bees turn back and look? *Journal of Comparative Physiology*, **172**, 549–563.

Mallot, H. A., & Franz, M. O. 2000. Biomimetic Robot Navigation. *Journal of Robotics and Autonomous Systems*, **30**, 133–153.

McNaughton, B.L., Barnes, C.A., Gerrard, J.L., Gothard, K., Jung, M.W., Knierim, J.J., Kudrimoti, H., Qin, Y., Skaggs, W.E., Suster, M., & Weaver, K.L. 1996. Deciphering the hippocampal polyglot: the hippocampus as a path integration system. *Journal of Experimental Biology*, **199**, 173–185.

Poucet, B. 1993. Spatial cognitive maps in animals: new hypotheses on their structure and neural mechanisms. *Psychological Review*, **100**, 163–182.

Save, E., & Poucet, B. 2000. Involvement of the hippocampus and associative parietal cortex in the use of proximal and distal landmarks for navigation. *Behavioural Brain Research*, **109**, 195–206.

Schomaker, L., & Fehrmann, R. 2003. Learning an approximate map of the environment by unsupervised bimodal landmark exploration. *Proc. of the 15th Belgian-Dutch Artificial Intelligence Conference (BNAIC'03), Nijmegen, The Netherlands*, 275–282.

Shen, Jun-Xian, Xu, Zhi-Min, & Hankes, Elmer. 1998. Direct homing behaviour in the ant Tetramorium Caespitum(Formicidae, Myrmicinae). *Animal Behavior*, **55(6)**, 1443–1450.

Tolman, E.C. 1948. Cognitive maps in rats and men. *Psychological Review*, **55(4)**, 189–208.

Wehner, R. 1984. Astronavigation in insects. *Annual Review of Entomology*, **29**, 277–298.

Wehner, R., Michel, B., & Antonsen, P. 1996. Visual navigation in insects: coupling of egocentric and geocentric information. *Journal of Experimental Biology*, **199**, 129–140.