Faculty of Aerospace Engineering

Visual Homing for Micro Aerial Vehicles using Scene Familiarity

An Insect-Inspired Approach.

G.J.J. van Dalen

May 19, 2016

TU Delft Delft University of Technology

Challenge the future

Visual Homing for Micro Aerial Vehicles using Scene Familiarity

An Insect-Inspired Approach.

Master of Science Thesis

For obtaining the degree of Master of Science in Aerospace Engineering
at Delft University of Technology

G.J.J. van Dalen

May 19, 2016

Faculty of Aerospace Engineering · Delft University of Technology

Delft University of Technology

Delft University Of Technology
Department Of
Control and Simulation

The undersigned hereby certify that they have read and recommend to the Faculty of Aerospace Engineering for acceptance a thesis entitled **"Visual Homing for Micro Aerial Vehicles using Scene Familiarity"** by **G.J.J. van Dalen** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: May 19, 2016

Readers:

Dr. Q.P. Chu

Dr. G.C.H.E. de Croon

ir. K.N. McGuire

Dr. ir. C.J.M. Verhoeven

Preface

Verdwaald in het bos,
Blik ik terug
Op alles wat achter me ligt;
De obstakels,
maar ook de openheid van mijn zicht,
Om roerloos hierdoor
te blijven stilstaan
En me af te vragen waar
Naartoe te gaan.
Om doelen te stellen voor
Een weg vooruit
En te omvatten wat
Alles voor me beduidt.
Door deze inzichten van
Het verleden en het heden
Weet ik me nu in
Een bepaalde richting
Te begeven.

Leen Aarts

In front of you lies the thesis I have been conducting over the last several months. It is the final work I deliver, in order to meet the graduation qualifications for the Aerospace Engineering masters at Delft University of Technology, the Netherlands.

During my internship at Georgia Institute of Technology (Georgia Tech), Atlanta, USA, I developed an absolute localization algorithm for Unmanned Aerial Vehicles (UAVs). During this internship I decided to select a thesis topic in UAV Guidance Navigation & Control (GNC).

I asked Guido de Croon to be the supervisor for my thesis work. Guido let me choose between different thesis topics, after which I chose to work on high-level insect-inspired visual naviga-

tion. This gave me the opportunity to work with other research disciplines than engineering and it forced me to take computational constraints of micro UAVs into account.

I would like to thank Kimberly and Guido, for their tips and feedback on my research. Kimberly was a great sparring partner, who helped me stay motivated throughout the process. Combined with Guido's ample experience and relaxed attitude I am very grateful with these supervisors. Furthermore, I thank members of MAVLab for help and motivation during the project. In particular Kirk Scheper facilitated me a lot during my struggles with the SmartUAV simulator. Also, I would like to thank Tobias Heil for proofreading the work. Last but not least, I would like to thank friends and family for listening, when I rambled on about robotics, programming or artificial intelligence.

With the poem above (in Dutch), I would like to finish this personal note, by symbolizing the thesis and showing the transition I am going to make from the academic to the industrial world.

Gerald van Dalen
May 19, 2016
Delft, The Netherlands

Acronyms

ALV	Average Landmark Vector
BoA	Basin of Attraction
BoW	Bag of Words
EKF	Extended Kalman Filter
GNC	Guidance Navigation & Control
GPS	Global Positioning System
GVG	Generalized Voronoi Graph
HSV	Hue Saturation Value
IMAV	International Micro Air Vehicle Conference and Competition
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
LK	Lucas-Kanade
MAV	Micro Aerial Vehicle
MP	Milestone Position
MPU	Micro Processing Unit
NN	Neural Network
PI	Path Integration
PTaM	Parallel Tracking and Mapping
RANSAC	Random Sample Consensus
RMS	Root-Mean-Square
SfM	Structure from Motion
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SSD	Sum of Squared Differences
SVO	Semi-direct Visual Odometry
UAV	Unmanned Aerial Vehicle
VO	Visual Odometry

List of Definitions

Active Navigation In the context of this thesis it is the opposite of exploration; i.e., it can entail navigation to an earlier visited location.

Exploration Journey The journey where a UAV flies around without navigating. In the context of homing this often means it is used for route learning.

Fingerprint Any representation of a mapped place of interest in the environment (i.e., node).

Homing The proces of returning to an earlier visited location. It can be used for animals or unmanned vehicles.

Homing Problem Part of the navigation problem, which only considers returning to an earlier visited location.

Homing/Return Trajectory Path which is followed to reach an earlier visited location.

Landmark An object or feature of a landscape that is easily seen and recognized from a distance, especially one that enables someone to establish their location.[1]

Low-Level Autonomy Flying behavior on a lower cognitive level than homing navigation, like stabilizing, turning or obstacle avoidance.

Map-Based Navigation Navigation where either a pre-defined map is used, or a map is generated and used for navigation afterwards.

Mapless Navigation Navigation where no map is used.

Navigation The process of determining and maintaining a course or trajectory to a goal location.

Navigation Problem The broad definition of navigation, consisting of self-localization and navigation within the environment.

Visual Homing Homing using vision as main sensory input.

Visual Odometry Integration of velocity extracted from optic flow, in order to get a location estimate. In this thesis this estimate can be aided by information from other sensors.

[1] http://www.oxforddictionaries.com/definition/english/landmark

Contents

Chapter 1

Introduction

One of the major scientific challenges with regard to research to Micro Aerial Vehicles (MAVs) are the limitations concerning both sensor and computational capacity of the vehicles. Especially when GNC tasks are required to be done on-board these very limited platforms, clever sensor usages and algorithm designs must be formulated.

A well-known use case for MAVs is performing vision-based exploration missions. Two examples of this are exploration of disaster environments and stock checking in warehouses. Low-level requirements for these missions include the ability to move, change direction and avoid obstacles. Much research has already been done in these fields.

Another element is the ability to navigate actively. In the context of exploration missions this active navigation can entail the ability to return to an earlier visited location or more specifically the initial location. Especially if environments are unknown (i.e., no external sensors available) and unreachable by humans, it is important to return to this initial location.

The navigational ability to return to the initial location is in literature referred to as *homing*, derived from homing behavior of insects (Nelson, 1991). An example of this homing behavior has been observed in desert ants (*Cataglyphis fortis*), where path integration mechanisms are employed to find straight paths to the nest (Muller & Wehner, 1988). Given the limitations in both sensor- and cognitive capabilities of ants, the homing strategy employed must be simple and yet powerful.

Translating homing to MAV research, insect behavior can be mimicked by very small vehicles with extremely limited computational resources. Especially for situations where both sensing and computing must be performed fully on-board, this problem is not yet solved.

When vision is the driving sensor, home-bound navigation is referred to as *visual* homing. Application of this is observed in both bee- and ant navigation. In case of UAV implementations, this means the main sensor used is a camera.

The goal of this thesis is to investigate a recently published method on insect behavior, where solely the visual familiarity of a route is used to perform visual homing (Baddeley, Graham, Husbands, & Philippides, 2012). The focus is on investigating the biological proof of concept

presented by Baddeley et al., to find out whether the algorithm is suitable for implementation on MAVs.

1-1 Research Context & Relevance

The research done by MAVLab[1] at TU Delft has been and still is focused on low-level flight and (vision-based) autonomy of MAVs. Notable examples concerning DelFly are system identification (Caetano et al., 2013), vision-based flight (de Wagter, Tijmons, Remes, & de Croon, 2014; G. de Croon, de Clercq, Ruijsink, Remes, & de Wagter, 2009), obstacle avoidance and autonomous flight in general (G. C. H. E. de Croon et al., 2012, 2013).

A missing element on these MAVs is autonomous navigation with on-board sensing and on-board processing only. This thesis should be a first step in implementing vision-based navigation algorithms on-board micro aerial systems. Especially the ability to navigate back to the initial location is a vital skill for a drone to have in an exploration mission.

1-2 Thesis Layout

The main part of this thesis is a scientific article, containing the key methods, results, conclusions and recommendations. This article is presented in Chapter 2 of this report. A more concise version of this paper is submitted to the International Micro Air Vehicle Conference and Competition (IMAV) 2016 and can be found in Chapter 3.

As addendum to this article, several chapters are added to give more context and show earlier research done. First, Chapter 4 contains an overview of the literature reviewed in an early stage of the project. Secondly, Chapter 5 shows a detailed explanation of the Infomax neural network: an unsupervised neural network used for route learning. This neural network is applied in closed-loop simulations presented in the scientific papers. Finally, Chapter 6 contains extra details regarding the simulation environment used to obtain the results presented in the papers.

[1] `http://mavlab.tudelft.nl/`

Chapter 2

Thesis Article

This chapter contains the scientific article which is the main body of this thesis. It can be read on its own.

Visual Homing for Micro Aerial Vehicles using Scene Familiarity

Gerald J.J. van Dalen* Kimberly N. McGuire‡ Guido C.H.E. de Croon†

***Abstract*—Autonomous navigation is a major challenge in the development of MAVs. When an algorithm has to be efficient, insect intelligence can be a source of inspiration. An elementary navigation task is homing, which means autonomously returning to the initial location. A promising approach makes use of visual familiarity of a route to determine reference headings during homing. In this article an existing biological proof of concept based on desert ants is transferred to MAVs. Vision-in-the-loop experiments in different environments are performed, to investigate the viability of scene familiarity for visual navigation. Trained images are used to determine which control actions to take during homing. To determine familiarity, either a database of stored images is kept or an artificial neural network is used. Different image representations are compared in multiple simulated environments. The use of textons for determining familiarity gives the best performance, but HSV color histograms also perform well and are very efficient. It is concluded that to make this method competitive with other visual navigation approaches, route familiarity should be combined with other methods to improve robustness.**

I. Introduction

A major challenge in robotics is to navigate autonomously through an unknown environment. Especially in indoor scenes, where no Global Positioning System (GPS) system is available, the entire navigation problem has not been solved yet.

Current navigation algorithms either require expensive sensors or significant computation power. Examples include Simultaneous Localization and Mapping (SLAM) methods, which have shown to be successful in real-time navigation, on platforms with enough power. Most Micro Aerial Vehicles (MAVs) do not carry such sensors and cannot perform heavy computations on-board the vehicle.

In order to find suitable navigation algorithms for MAVs, insects can be a source of inspiration, since they constantly have to deal with complex navigation tasks while only having small-sized brains [1]. Different algorithms have already been created based on observations done on insect. A well-known example is the use of optic flow to get a sense of velocity, which is known to be done by insects [2]. Integrating this estimate for localization is called visual odometry. The obtained location estimate is employed in higher level navigation algorithms. Still, these algorithms are not readily available for tiny MAVs yet. One of the higher level skills employed by insects is the ability to return to the nest location. This is referred to as *homing* [3]. It would be an important enabler for MAVs, if they could use similarly high-level, but computationally efficient algorithms for navigation.

A promising homing algorithm is proposed by Baddeley et al., where familiar views along a route are used to determine the correct direction to an earlier visited location [4]. This is a *visual* homing algorithm, since cameras are used as driving sensor. Instead of focussing on the construction of a detailed (or coarse) map, Baddeley et al. propose that homing can be performed just by means of recognizing which direction seems most familiar to a robot. Furthermore, they use a small neural network to store and recapitulate a route in order to find the initial location. Potentially, this is very useful for MAV navigation algorithms, since it deals with limited memory found on many small platforms.

In an effort to find efficient navigation algorithms for MAVs, this paper investigates the practical application of the scene familiarity algorithm. The focus is on how robust familiarity is to determine control actions. First, section II discusses the state-of-the-art in autonomous visual navigation on drones. Then, section III explains the scene familiarity method as introduced by Baddeley et al. After identifying the limitations in the simulations presented, sections IV, V and VI show simulations and experiments for different environments, to overcome current shortcomings in the implementation described by Baddeley et al. Finally, closed-loop simulation flights are performed to show a more realistic use-case of view familiarity for MAV homing.

II. Related Research

The scene familiarity method introduced by Baddeley et al. is a biologically plausible method to find a solution for the visual homing problem [4]. Visual homing is in principal an element of the general navigation problem, to which much research is being done. Many algorithms made for solving this navigation problem can potentially be used to solve the homing problem. In this paper, the following definition for navigation is used: *"Navigation is the process of determining and maintaining a course or trajectory to a goal location."* [5]. This basically states that the aim is to navigate to a goal location, without enforcing to have knowledge about the current location of the robot. This section gives a brief overview of previous research done to visual navigation and specifically visual homing.

*Gerald van Dalen is a graduate student at the Micro Air Vehicle Laboratory at TU Delft. Email: gjj.vandalen@gmail.com

‡Kimberly McGuire is a PhD candidate at the Micro Air Vehicle Laboratory at TU Delft.

†Guido de Croon is an assistant professor at the Micro Air Vehicle Laboratory at TU Delft.

A. Map-based Navigation

Map-based navigation refers to methods where a map is kept to navigate. Since the navigation algorithms we are interested in must work in different, priorly unknown locations, only SLAM methods are considered [6].

SLAM is a paradigm in which the agent generates a map when traversing a route, and simultaneously localizes itself and navigates on this map. This means that SLAM in principal consists of three parts, namely traversing/exploring a route, constructing a map and self-localization and navigation in this map [7]. This means, map-based methods require information about the current location of a robot. To make generated maps more robust, loop closure is performed, where errors are corrected by re-visiting mapped locations [8], [9], [10].

Since SLAM is a general navigation method, homing can also be done with it. This is for example shown in Motard et al., where an AIBO robot[1] must navigate back to its charging station [11]. Still, running SLAM, and in particular visual SLAM algorithms in real-time, requires much computational power, since (visual) processing, mapping and self-localization must be performed simultaneously. Since most MAVs have limited computational power, visual SLAM often cannot be run in real-time, which makes it less suitable for homing on small platforms.

B. Path Integration

Path Integration (PI) methods form navigation solutions where localization at different moments in time are used to navigate. A velocity estimate is obtained and integrated to perform localization. In principle, no map is made of the environment, although when heading estimation is part of the localization, the result will be a map-like representation.

In visual driven robotics, velocity is often obtained using optic flow between different frames of a scene [12]. Often, a downward looking camera is used to obtain images. Optic flow is the apparent motion observed in an image and many different algorithms exist to calculate optic flow. A comparison of these methods and different applications to Unmanned Aerial Vehicles (UAVs) are published by Barron et al. [13] and Chao et al. [14] respectively. For UAV applications, the most common algorithms for calculating optic flow are the Lucas-Kanade (LK) method [15], the Horn-Schunck method [16], image interpolation methods [17], block matching techniques and feature matching techniques.

For the homing problem the aim is to record and store the traversed route of an MAV. As mentioned in the introduction, integrating optic flow to estimate location is called visual odometry. Research in visual odometry is done for both stereo vision and monocular vision [18]. When stereo vision is used, it is easier to make a three dimensional representation of the environment due to available depth information. For monocular vision, this depth information must be extracted from sequential images. This means that stereo visual odometry can operate without motion, where in the monocular case motion is needed to extract environmental geometry.

[1] http://www.sony-aibo.co.uk/

C. Snapshot Model

One way in which insects can store a location is by making a snapshot. This means a location is marked by storing an image. When performing localization, the current visual input is compared to this snapshot in order to converge toward the location. In 1983, Cartwright & Collett introduced the Snapshot Model [19]. The framework they presented explains the navigation capabilities of bees when traveling between different food sources. The model consists of two main elements: a dead-reckoning method to get close to the goal location and finding the best visual match with a stored snapshot to find the exact location of this goal. The visual matching is done by a direct comparison of an image on the retina with a stored snapshot.

For highly cluttered environments close to the goal location, the visual matching would only work when the distance is very small. Cartwright & Collett extended the snapshot model by adding an extra snapshot, which does not contain landmarks close to the goal [20]. This snapshot can be used for visual matching when the distance to the goal is larger, while the other snapshot (including visual information close to the goal) can be used for the last part of the homing route. Navigation based on those snapshots is done by comparing size and azimuth of the landmarks between the snapshot and retinal image [21].

The landmark approach is further extended with the addition of visual beacons along the route [22]. During the early parts of the return to a feeder or nest location, not only dead-reckoning, but also visual landmark information is used. The usage of those beacons reduce the occurence of integration errors. A disadvantage of this is that many images have to be stored.

In order to make the snapshot model biologically more plausible, a neural implementation is developed [23]. It gives an explanation of why and how the snapshot model can work in an insect's brain. The paper shows that even though the real neural implementation in an insect's brain is unknown, a simple neural model can mimic snapshot homing.

D. Average Landmark Vectors

The snapshot model stores an image to represent a certain location of interest, like a nest or food source. From this, a heading vector to the home location is obtained. A similar approach uses Average Landmark Vectors (ALVs) to represent landmarks [24]. ALVs, introduced by Lambrinos et al. in 1998, are averages of the heading vectors to all landmark locations [25]. The homing vector is determined with respect to this ALV. Lambrinos et al. classified objects as landmarks using a brightness threshold on pixel intensities [24]. When a patch of pixels above (or below) this threshold is available, it is recognized as a landmark.

The main difference between ALV homing and snapshot homing is that a location of interest is represented by a single

vector, instead of an entire image. This both improves the computational efficiency and reduces the storage demands. The downside is that it is less accurate and prone to errors. A small offset of a landmark vector can have a big impact on the ALV and thus the homing vector. Furthermore, the current heading of a robot is needed to be able to rely on ALVs.

E. Scene Familiarity

Scene familiarity methods refer to recognition of a traversed route, without specific information about the goal location. This means, a robot must always move into the *most familiar* direction. In the ideal case, this would automatically mean that the agent returns to the goal location.

In 2012, Baddeley et al. proposed a scene familiarity method for visual homing of desert ants [4]. The purpose of this research was to give a proof of concept for how desert ants, which do not rely on pheromone trails, might navigate back to their nests.

This scene familiarity method is quite new and to our knowledge not yet used in robotic applications. Our paper analyzes the method as a first step towards applying the algorithm on-board MAVs. The focus is on computational efficiency of the algorithm. The next section reviews the paper published by Baddeley et al. in depth.

III. The Scene Familiarity Method

In an effort to find a biologically more plausible alternative to map-based navigation methods and the snapshot model described in the previous section [26], the scene familiarity homing method is introduced [4]. To show that homing navigation could take place without the use of visual odometry, a method is presented where views along the entire route determine the heading in which to proceed. Conceptually, this means that during a training run images in the direction of the route are stored. Then, when using the algorithm for homing, images taken around the robot are compared to these stored views in order to determine the most familiar direction.

The method is both presented assuming a *perfect memory*, as well as a biologically more plausible neural method. The first method stores pictures taken during a training run and matches them for homing using the Sum of Squared Differences (SSD) of raw pixel values. The second method uses an unsupervised neural network called Infomax to approximate familiarity [27]. The network can be seen as an *approximator* of the familiarity of images that would otherwise be stored. Next to the biological likeliness of such a network, it gives control over the required computational power.

In this section, first homing using both a perfect memory (section III-A) and an Infomax neural network (section III-B) are summarized. Then, in section III-C an overview of issues is given, which must be addressed before the scene familiarity method can be applied on-board an MAV.

A. Route Representation using a Perfect Memory

The algorithm presented by Baddeley et al. makes use of an exploration (training) run and a homing (testing) run. During exploration, path integration and obstacle avoidance are used to progress through the environment. This is the training phase of the agent, in which a perfect memory is used for storing views. This entails storing a panoramic image every $4cm$. The panoramic images serve as *visual compass*, since they have a known field of view of $360°$ and are centered in the direction of travel of the agent.

Fig. 1. Binary panoramic image used in Baddeley et al. [4].

When the homing capabilities are tested, the agent is placed back at its initial location. From there, homing is done by taking $360°$ images with an omni-directional camera. Each panorama is iteratively shifted by 1 pixel, such that the center of the image is placed in all direction. These shifted panoramas are matched against the stored database of images, to find the most familiar view. This most familiar view is found by maximizing the familiarity values between each shifted panorama and each stored image. The familiarity value between a single view and all stored images is obtained by calculating the SSD of raw pixel values, as defined in Equation 1 [28].

$$F(I) = -\arg\min_{i} \sum_{x,y} \left(I(x,y) - V_i(x,y)\right)^2 \qquad (1)$$

In this equation, $F(I)$ is the familiarity of view I, $I(x,y)$ is a single panoramic view and $V_i(x,y)$ are the stored views. It can be seen that the stored image that gives the closest match to the current image is used as familiarity value. Note that the agent can rotate on the spot or use an omni-directional camera to obtain familiarity values in all directions. The latter approach is used by Baddeley et al. After determining the most familiar direction (by maximizing the values obtained with Equation 1), the simulated agent is moved in that direction by $10cm$. After this the procedure is repeated.

In the paper, the stored panoramas are binary images and have dimensions of 90 by 17 pixels (Figure 1). The resolution is such that each pixel in horizontal direction is equivalent to a rotation of $4°$. During homing, familiarity is evaluated for steps of 1 pixel, such that each panorama is shifted 90 times and hence Equation 1 is evaluated 90 times. As said, the maximum outcome of these 90 familiarity values results in the most familiar direction.

An example of the simulation environment is shown in Figure 2. Simulations using a perfect memory are done with different clutter densities in the environment. The presented results show that densely cluttered environments are harder to recognize than sparsely cluttered ones. This can be explained by obstruction of views; especially when objects appear in the vicinity of the agent, scenes can differ significantly by only moderate displacements of the agent. Figure 2 also shows the results presented in Baddeley et al. for a perfect memory. Each

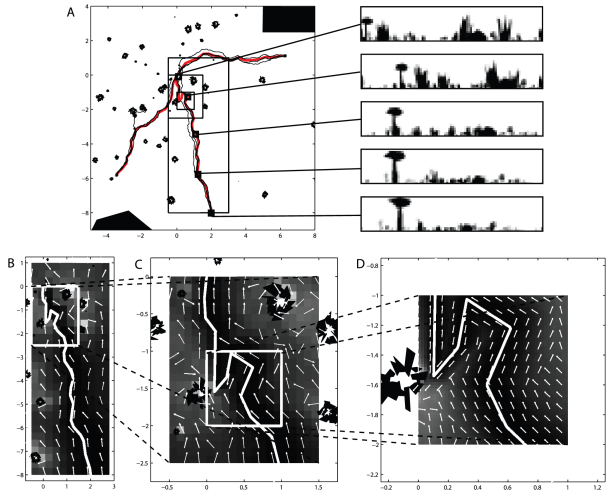


Fig. 2. Figure A shows the simulation environment used by Baddeley et al., including typical views experienced along the route [4]. The three red lines show the learned routes and the black lines show different homing runs, performed using a perfect memory. Figures B, C and D show parts of the route at different scales.

of the three routes contain between 700 and 980 stored views. During homing, Gaussian noise with a standard deviation of approximately 5° was added to the heading during each step performed by the agent. This is done to simulate uncertainties during navigation.

Looking at the results, it can be concluded that the simulations show very good performance. Due to the large memory and computational requirements, the algorithm in the current form is not yet suitable for implementation on-board a small robot. The next section shows the use of a neural network for the storage of familiarity, to reduce these computational requirements.

B. Route Representation using an Infomax Neural Network

In their paper, Baddeley et al. also study a neural network approach for determining familiarity. The goal of the homing method is not to recapitulate the entire route, but only to get a sense of familiarity. Using a neural network limits the storage requirements for the route, since all information is encoded in the initially defined neurons.

To store the familiarity of a route, an Infomax neural network is chosen [27], [29]. The Infomax neural network used consists of two layers, with an input layer and a novelty layer (Figure 3) [27]. The network can be used for both feature extraction and familiarity discrimination. However, for the method proposed by Baddeley et al., only familiarity discrimination is needed. This familiarity discrimination works as follows. Each input provided to the network as training sample changes the weights such that the input to the second layer (the novelty layer) is lowered. This means when during testing familiar samples are provided, the summed input to novelty neurons is lower than for unfamiliar samples. Note that in our paper, a higher outcome means more familiar; therefore, a minus sign is added to the summed novelty layer input.

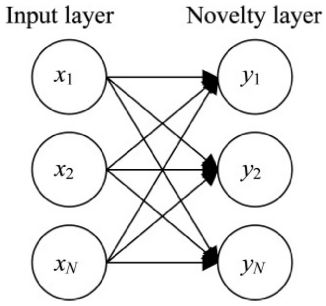


Fig. 3. Infomax neural network structure with an input layer and a novelty layer. In this representation it is assumed that the input layer and novelty layer contain an equal amount of neurons. Obtained from [27].

As input to the network raw pixel values of filtered binary images are used. The number of input neurons is equal to the number of novelty neurons. In principle this is not a given necessity, since a lower amount of novelty neurons is computationally advantageous and might give sufficient performance for successful scene discrimination. On the other hand, a higher amount would increase the storage capacity of the network [27]. N is used to indicate the number of inputs, while M indicates the number of novelty neurons.

As mentioned above, the main idea behind an Infomax network for familiarity discrimination is that any sequence of inputs encountered during training adjust the weights such that the total input to the novelty layer decreases. The metric for familiarity is defined as:

$$d(x) = -\sum_{i=1}^{M} |h_i| \tag{2}$$

Here, $d(x)$ (also called the *decision function*) is the familiarity of input sequence x, for which a larger value means that the sequence is more familiar than when $d(x)$ is smaller. h_i is the input to the ith novelty neuron and is defined as:

$$h_i = \sum_{j=1}^{N} w_{ij} x_j \tag{3}$$

In this equation x_j is the input from the jth input neuron. Finally, the activation function of the ith novelty neuron is a hyperbolic tangent of h_i.

As the familiarity $d(x)$ can be seen as the desired output of this network, an output layer is not needed and therefore discarded.

Training is done using an unsupervised learning rule, with the aim to lower the familiarity for each sample encountered during training. The difference between supervised learning (which is normally used in neural networks) and unsupervised learning, is that in supervised learning the difference between desired and actual output of the network is minimized to update the weights of the neurons. In unsupervised learning, however, the desired output is not used for training. Instead, an update rule as function of network input, novelty layer output

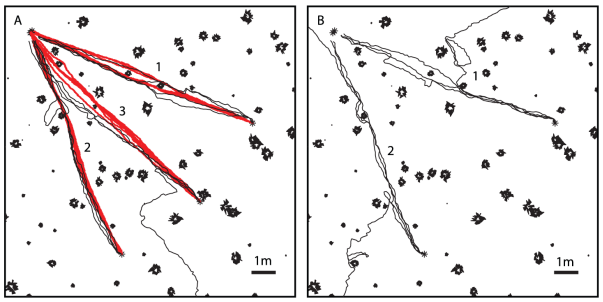


Fig. 4. Multiple routes trained using the same Infomax neural network. The numbers indicate the order in which routes were trained. Obtained from [4]. In Figure A, homing on each route is performed immediately after training. In Figure B, familiarity on routes 1 and 2 is evaluated after having finished training all routes.

and current neuron weights is applied to update the weights. On the one hand this makes unsupervised training very fast, but on the other hand it provides less control over the trained output of the network. As the familiarity output of an Infomax network is only used to compare the result with respect to other inputs, unsupervised learning suffices. The unsupervised learning rule used is obtained from [30] and is defined as:

$$\Delta w_{i,j} = \frac{\eta}{M}\left(w_{i,j} - (y_i + h_i)\sum_{k=1}^{M} h_k w_{k,j}\right) \qquad (4)$$

In this equation, η is the learning rate, $w_{i,j}$ is the current value of the weight between input j and neuron i and y_i is the output of the ith novelty neuron.

Baddeley et al. also presented simulation results using the Infomax network as storage for views. The only difference is the use of panoramic images of $180°$ instead of $360°$, centered on the heading chosen in the previous timestep. The results using this network show to be very similar to the results using a perfect memory. Therefore, these are not included here.

When dealing with artificial neural networks, providing new training data will eventually cause the network to forget earlier provided views. To investigate this, Baddeley et al. attempted to learn three different routes sequentially, using the same network. The results of this are shown in Figure 4. It can be seen that training three routes (i.e., a total learned distance of approximately 30 meters) increases the failure rate during homing.

C. Issues for Robotic Implementations

After having presented a brief overview of the results obtained by Baddeley et al., the following lists the issues that must be addressed before being able to implement the algorithm on a robot:

- **Environment:** Binary sceneries are used: the sky is white and objects are black. These environments are not representative for the scenes through which a robot must navigate. The reasoning for using such environments, is that it may be representative for the views experienced by ants. Even though these binary images are not representative for a real environment, it might be possible to extract such a *skyline* from normal camera images. Experimenting with different, more realistic environments will be the main focus of this paper.
- **Scanning accuracy:** For determining familiarity, forward looking panoramic images are used. The simulation is set up such that moving the image by one pixel in the horizontal direction is equivalent to a rotation of the agent of $4°$. In the simulations, the most familiar direction is chosen by comparing different views with the stored panoramas. The different views are a single pixel (hence, $4°$) apart. These accurate rotations are not realistic for application on an MAV.
- **Computation:** The simulations using a perfect memory store 700 to 980 views per route. To perform a scan over $360°$, 90 views have to be matched with each stored images. This comes down to a total number of image comparisons between 63000 and 88200 per timestep. This was solved by using an Infomax network instead, where approximately 1500 input neurons and novelty neurons are used. This amount of novelty neurons makes it unusable in real-time, but a smaller number may lead to similar performance and better computional efficiency. Comparing the real-time performance of different image representations is implicitly done during the closed-loop simulations, since they are performed in real-time. For the Infomax network this means less novelty neurons are used. Also, the views used are scaled down to decrease the number of inputs to the network.
- **Short distances:** The simulation results clearly show the applicability to insects. This is seen in the fact that images are stored every $4cm$ and movements of $10cm$ per timestep are made during homing. When the method is implemented in robotics, the robot should be able to cover longer distances to make it more useful. This might not be a problem, but has to be tested before concluding the usefulness on UAVs.
- **Training run:** Training of the route is done in the direction of homing. Ultimately, it is desired to perform training during exploration, while homing occurs in the opposite direction. There are several ways to circumvent this, like adding an extra camera on the back of the robot, using an omni-directional camera or by making a turn when homing is initiated (i.e., *navigation behavior*). When the goal of the algorithm is to run on platforms with a single, forward looking camera, this must be further investigated. In the closed-loop results presented in our paper, the simulated MAV flies backwards during training and forward during homing.

Recently, Gaffin et al. have published a detailed analysis on scene familiarity in realistic, indoor environments [31]. Distinguishing familiarity is both analyzed in rotation and translation, for raw pixel matching between images of different resolutions. A rail mounted camera is used to perform

MATLAB-driven experiments.

In our analysis of the scene familiarity method, we will use a simulator containing realistic sceneries, vehicle dynamics and camera parameters. A translation and rotation analysis will be performed as well, however, next to raw pixel values, we will also investigate alternative image representations, to determine which one is more suitable for recognizing familiar views. Closed-loop simulations with an MAV are presented and we show the use of an Infomax neural network as well, since this helps in meeting the limited storage requirements of an MAV. We hope to better understand autonomous navigation for small MAVs.

IV. Environment Analysis

In the previous section, the original simulation results presented by Baddeley et al. are discussed. Based on this, a key question remains whether the algorithm will work in different types of environments. As mentioned, only filtered binary images were used as experienced views. Also, the SSD between pixel values of two views is used for familiarity registration, while there may be more viable methods to evaluate this. Similarly, raw pixel values were given as input to the Infomax neural network, while other metrics may work as well or better.

In this and the following sections, an analysis of different environments is presented in combination with different image representations. First, this section gives an overview of the different image representations tested and introduces the different image matching performance criteria. Then, section V shows simulation results of these different methods in multiple environments and section VI shows similar results for real imagery, in an attempt to validate the simulations. Note that the focus is not on the use of a neural network for storing familiarity, since this (and other approximators) can be applied to either image representation.

To test the usability of familiarity of scenes for visual homing, we investigate the familiarity sensitivity to both rotation and translation. Analyzing rotation is done by performing a 360° turn at a fixed location in the environment, in steps of 5°. A single view is stored and used as trained image and all other views experienced during this rotation are compared to it. The hypothesis is that familiarity should improve when the heading difference between the current view and the stored image decreases.

Translation is analyzed by evaluating familiarity in a grid of locations, with a fixed heading. Again, a single view is used as training sample and the familiarity is expected to improve when the distance to the trained image gets smaller. Results of this should show the sensitivity of familiarity with both increasing distance (in two directions) and increasing heading angle. Rotation is considered to be most important, since heading commands are given to determine control actions during homing. Analysing translation is mainly done to determine how much familiarity changes when an agent drifts away. When these changes are significant, it is likely that the home location is not found when the robot does not follow the exact training path. Note that opposed to the results shown by Baddeley et al., a more realistic camera model is used instead of cropping parts of the environment.

The following image representations are compared in the different environments:

- **Raw pixel values** The sum of squared differences of each pixel in two images outputs a similarity score [28]. A value of zero indicates that the two images are identical and when the value gets larger, images are less similar. SSD on these pixel values is computationally expensive, since all pixels are compared. This method is used in the simulations presented by Baddeley et al. [4]. As mentioned, in this paper the score is inverted such that the most familiar view gives the highest score.
- **Texton histograms** Textons are small texture descriptive image patches, which can be extracted from an image [32]. The patches can be assigned to pre-trained texton clusters, such that a histogram is created which represents the image. This makes it less sensitive to small displacements compared to spatially variant image matching methods such as SSD between raw pixel values. An example texton histogram is shown in Figure 5. Image patches of 5 by 5 pixels are extracted from an image of a sports hall (as shown in Figure 5a) and classified to a dictionary of 50 textons (or: clusters). The patches in the dictionary are shown in Figure 5b. The patches extracted from the image are clustered by minimizing the euclidean distance with the textons in the dictionary, resulting in a histogram as shown in Figure 5c. In this paper the histogram frequencies are normalized; each cluster is divided by the total number of patches extracted.
- **Hue Saturation Value (HSV) color histograms** Color histograms contain a classification of each pixel based on color intensity. Here, HSV colors are used. A saturation threshold of 0.2 is used, which means all pixels with a saturation lower than this value are discarded. The corresponding Hue and Value channels of the pixels accounted for are used for the histogram. The Hue histogram contains 25 bins, as does the Value histogram. To obtain the HSV color histogram (which looks conceptually similar to Figure 5c), the histograms are concatenated.

The performances of the different methods in different environments are evaluated by 1) looking at how distinct a view close to the trained view is, compared to other views and 2) what the probability is that the correct (i.e., trained) view is selected as most familiar, since that direction will be chosen for homing. Figure 6 shows an example of a familiarity curve when rotating on the spot. The trained image is positioned at an angle of 180° and image matching is in this figure done using the SSD of raw pixel values. The performance is evaluated using the following measures:

- **Peak ratio** As mentioned, a rotation on the spot is used to compare all views around the robot with a single stored view. It is expected that the view that resembles the trained view best comes out as most familiar. If this

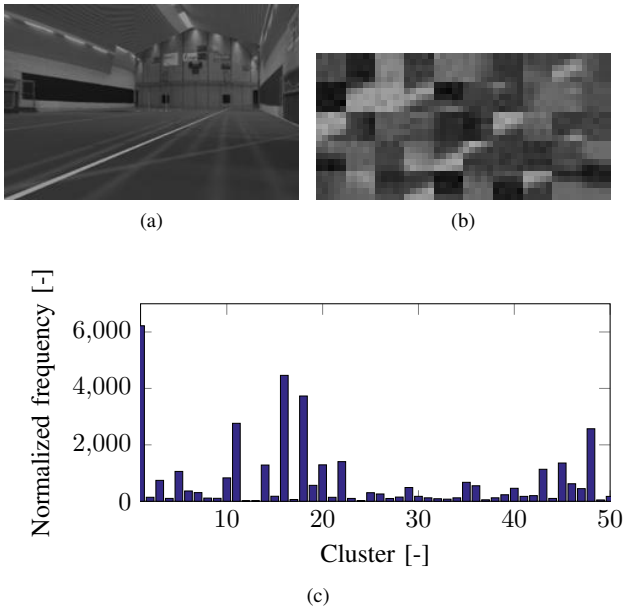


Fig. 5. Example of an image representation using textons. Figure 5a shows an example image from a sports hall. Figure 5b shows the clusters to which textons are assigned and Figure 5c shows the corresponding texton histogram. The textons are patches of 5 by 5 pixels, and a total number of 36816 textons have been extracted from the example image.

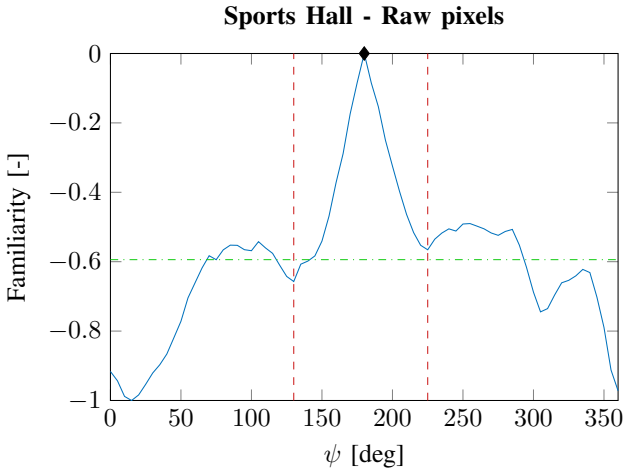


Fig. 6. Rotation on the spot at a constant location in the SmartUAV simulator. Unfiltered images of 48 by 32 pixels are taken every 5° and compared to a stored image at a heading angle of 180°. The red dashed lines indicate the BoA bounds and the green dashed line shows the mean familiarity.

is indeed the case, the distinctiveness of this peak can provide information on how likely it is that it will be picked as most familiar direction. In Figure 6 a green, dashed, horizontal line is drawn through the mean of the familiarity curve. This means the further the peak lies from the line, the more distinct a peak is.
To compare different environments, the peak ratio is defines as:

$$PR = \frac{\max F - \mu_F}{\max F - \min F} \tag{5}$$

In this equation, F refers to the familiarity values shown in Figure 6 and μ_F is the mean of all familiarity values (i.e., the green line in Figure 6). The higher the peak ratio is, the more distinct a peak is.

- **Basin of Attraction (BoA)** The basin of attraction shows how far an agent can be off from the trained view, before diverging away from the correct direction. It is evaluated by finding all local optima (both minima and maxima) and looking between which minima the agent converges towards the trained optimal familiarity (maximum). In Figure 6 two red, dashed, vertical lines are drawn through the local minima closest to the trained view. The percentage BoA is defined as the distance between those vertical lines divided by 360°. The larger this value is, the larger the probability that the most familiar direction is found.
- **Correlation coefficient** This is used to estimate the correlation between two neighboring heading angles, in this case differing by 5°. Here, the Pearson product-moment correlation coefficient is used, where 1 indicates full positive correlation between two neighboring angles, -1 means full negative correlation and 0 means no correlation.

The BoA is considered to be most important, since it determines how far an agent can be off the route (i.e., the correct heading) while still being able to converge back to the correct path, with a gradient-like search. The peak ratio is mainly useful when an agent has no clue where to go; if the agent makes a 360° turn and the the trained peak is very distinct, the probability of continuing in the right direction is high. The correlation coefficient gives a measure for how continuous a familiarity curve is. When the correlation is low, it could happen that spikes occur in the familiarity curve, which may give wrong results.

V. SmartUAV Simulations

This section analyzes different sceneries in the SmartUAV simulator. SmartUAV is developed for Guidance Navigation & Control (GNC) research to MAVs and specializes in the use of vision as primary sensor. The simulator is written in C++ and sensors and controllers can be connected using a block interface. This makes it easily extendable and the level of simulation fidelity can be adapted by changing complexity of vehicle dynamics, sensor dynamics and realism of the environment. Furthermore, the simulator can either run in real-time or fast-time.

Two different simulated environments are analyzed. Figure 7 shows example frames of each scene. As mentioned in the introduction, GPS can be used for outdoor navigation, which makes a visually driven homing algorithm less relevant. Therefore, indoor environments are analyzed, in contrast to what is done by Baddeley et al.

- **Sports hall** Figure 7a shows a frame from the TU Delft sports hall. The dimensions are 30 by 60 meters and the sports hall contains two orange poles on the center line of the hall.

(a) (b)

Fig. 7. Examples from the different sceneries used in SmartUAV simulations. Figure 7a shows a sports hall environment and Figure 7b shows a room with photos on the walls.

- **Photo room** Figure 7b shows an artificial gray room, containing randomly placed frames with photographs. The dimensions of the room are approximately 10 by 10 meters.

As mentioned, each environment will be tested on rotational and translational familiarity sensitivity. For familiarity estimation, SSD values of raw pixels, SSD values of texton histograms and SSD values of HSV color histograms are used and compared. The values are inverted and scaled for easy comparison.

A. Rotation

The familiarity sensitivity to yaw rotations is most important for view familiarity-based homing. Each turn taken during homing is made based on the familiarity values at different heading angles.

To analyze familiarity for different headings, this section presents performance for different image representations in the two environments. As explained in the previous section, the performance is evaluated by calculating the BoAs, peak ratios and correlation coefficients.

Rotation is analyzed by simulating an MAV on a single location in the environment and storing a representation of one view. Then, this representation is matched to views in all other directions in order to get a measure of familiarity. This is done in a grid of locations in each environment, to get both unobstructed views as well as obstructed views (e.g., close to a wall). The grids cover the entire sports hall and photoroom, and the spacings are 1 meter. For each location, the BoA, peak ratio and correlation coefficient can be calculated.

Table I summarizes these performance measures for the different methods and environments. The calculated BoAs, peak ratios and correlation coefficients are averaged for all locations and the standard deviations are included as well. Good performance is characterized by large BoAs (i.e., it is likely that the correct heading is found), large peak ratios (i.e., the correct familiarity value is distinct compared to familiarities in other directions) and correlation coefficients close to 1 (i.e., continuous and not too noisy).

Looking at the results, it can be seen that the BoAs for raw pixel matching and texton histogram matching are similar in both environments. HSV matching does not perform as good, which is also seen in lower correlation coefficients.

TABLE I
AVERAGE PERFORMANCE METRICS FOR EACH IMAGE MATCHING METHOD AND ENVIRONMENT COMBINATION DURING ROTATION.

	Raw pixels	Textons	HSV
Sports hall			
BoA average	37.3%	36.7%	6.90%
BoA std. dev.	16.5%	12.0%	3.77%
Peak Ratio average	0.57	0.43	0.53
Peak Ratio std. dev.	0.10	0.076	0.13
Correlation Coefficient average	0.98	0.98	0.80
Correlation Coefficient std. dev.	0.051	0.0091	0.14
Photo room			
BoA average	21.6%	22.0%	9.02%
BoA std. dev.	7.51%	8.89%	4.21%
Peak Ratio average	0.58	0.49	0.37
Peak Ratio std. dev.	0.092	0.12	0.089
Correlation Coefficient average	0.96	0.94	0.90
Correlation Coefficient std. dev.	0.023	0.026	0.037

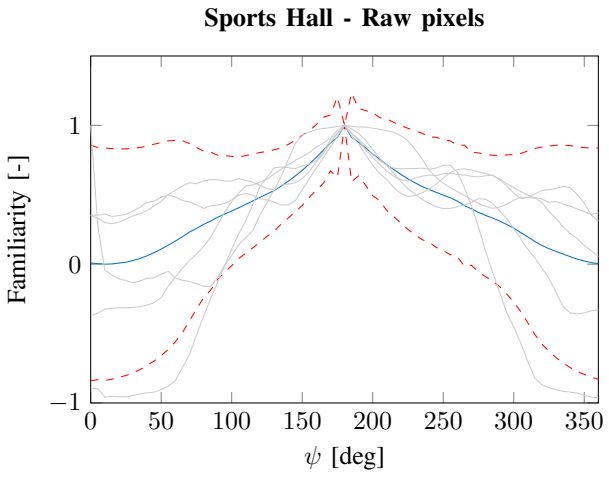
This indicates more local optima, which inherently reduces the BoA. The peak ratios are in both environments best with raw pixel matching. Since the values and standard deviations of these peak ratios lie close to each other for both environments, no significant conclusions can be drawn from this.

To illustrate the results shown in the table, familiarity curves of both environments are shown in Figures 8 and 9 respectively. Note again that the curves show how familiar a direction is, where a higher value means a better recognized view. The blue, solid lines indicate the average familiarity curve for all locations in the environments, the red dashed lines indicate two times the standard deviation and the gray lines show some example familiarity curves at individual locations in the environments. The results are scaled such that the average line lies between 0 and 1.

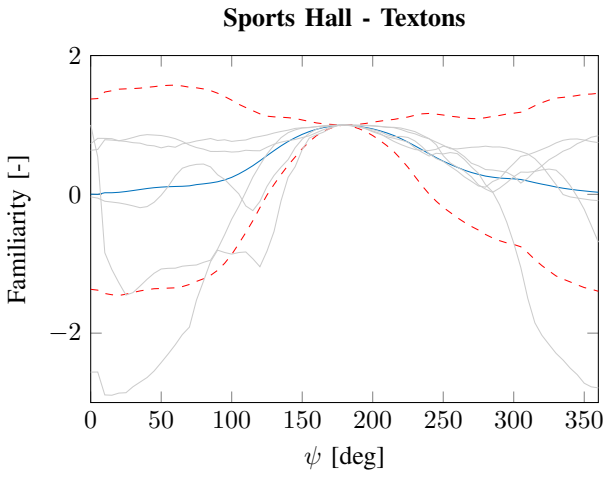
As expected, all average curves show a single peak at the trained locations (i.e., at $180°$). The HSV histogram result however, shows a less predictable outcome, with a larger amount of local optima. This was expected because of the lower BoAs and correlation coefficients shown in Table I.

When comparing the two different environments, it can be seen that the BoAs in the photoroom environment are lower than in the sports hall. An explanation can be found in the similarity in the photoroom environment, where each wall is gray and the only notable differences found are in the pictures on the walls. It seems that, at least for raw pixel and texton matching, a more realistic environments yields into higher BoAs. For HSV color matching this is the other way around, where the color differences found in the sports hall make HSV histograms more noisy. This especially leads to a lower correlation coefficient and peak ratio.

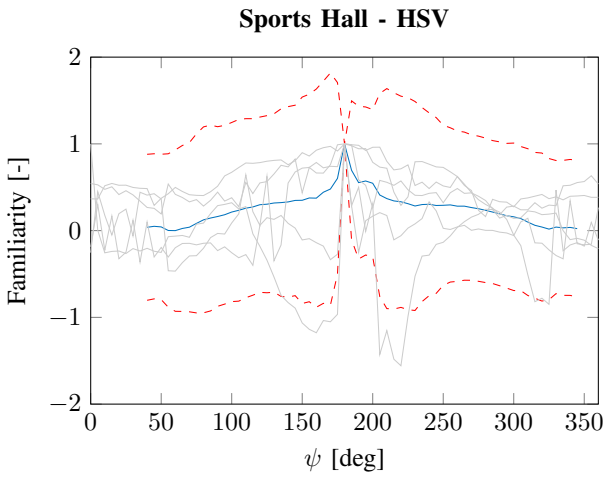
Because a BoA is very sensitive to the presence of local optima, smoothing the familiarity curves can lead to increased performance. When smoothing is applied, non-distinct local optima can disappear. Obviously, the more such a curve is smoothed, the more peaks disappear. When the right amount of smoothing is applied, the BoA can increase drastically, which makes it easier to converge towards the trained view. To test the effect of smoothing and the extent to which this



(a)

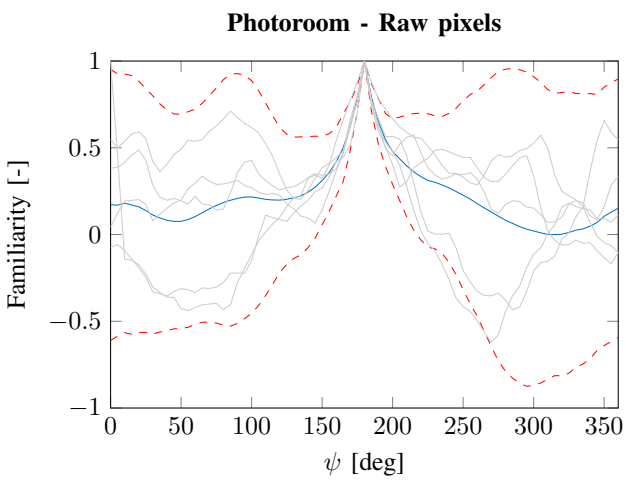


(b)

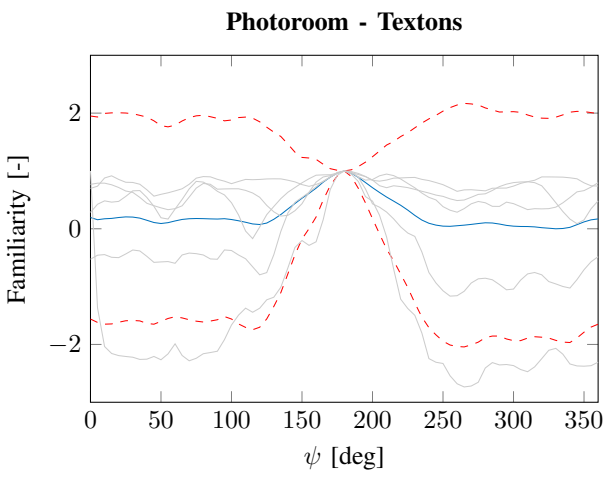


(c)

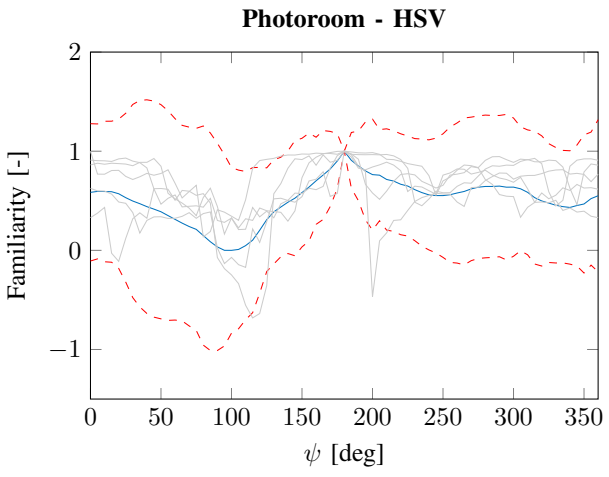
Fig. 8. Average rotation on the spot of 231 locations in SmartUAV. Unfiltered images of 48 by 32 pixels are taken every 5° and compared to a stored image at a heading angle of 180°. The red dashed lines indicate the 2σ bounds and the gray lines are some example familiarities.



(a)



(b)



(c)

Fig. 9. Average rotation on the spot of 81 locations in SmartUAV. Unfiltered images of 48 by 32 pixels are taken every 5° and compared to a stored image at a heading angle of 180°. The red dashed lines indicate the 2σ bounds and the gray lines are some example familiarities.

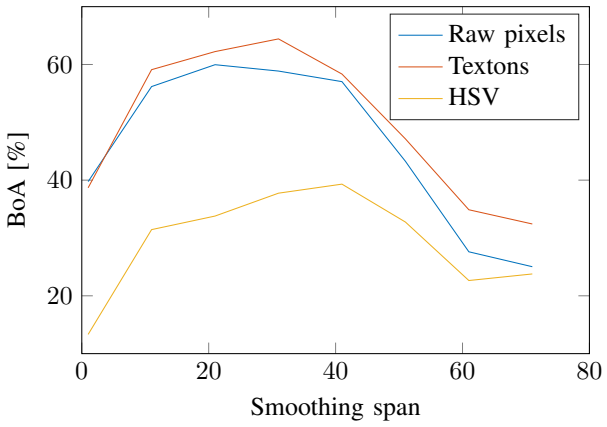


Fig. 10. Different BoAs for varying smoothness of the familiarity curve, evaluated in the sports hall environment. The x-axis indicates the smoothing span used for moving average smoothing. The curves are averages from all locations in the sports hall.

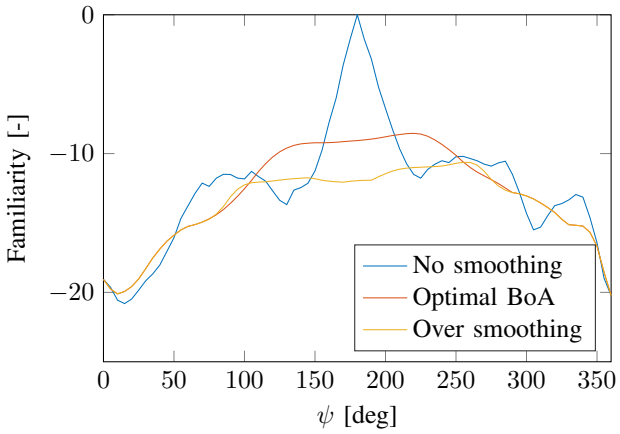


Fig. 11. Smoothed familiarities, for no smoothing, smoothing for optimal BoA (smoothing span of 30) and over smoothing (smoothing span of 70), in the sports hall environment.

is needed, a moving average filter is applied to the familiarity curves with varying span. Figures 10 and 12 show this for the two environments. Both plots show that smoothing (a moving average span of approximately 30°) drastically increases the BoAs of all three image representations. When too much smoothing is applied, the BoAs are lower, which indicate that the smoothing changed the shape of the familiarity curves too much. Figure 11 confirms this, by showing familiarity curves with different smoothing applied, using raw pixel matching. Note that smoothing a familiarity curve is not trivial, because the familiarity at all different headings must first be evaluated in order to apply smoothing. Then, a gradient-like search can be applied to find the most familiar direction.

In conclusion, raw pixel and texton matching show similar recognition performance in rotation. They both perform better than HSV matching, which suffers from the significant differences between color histograms at different angles. For both environments and matching methods, smoothing the

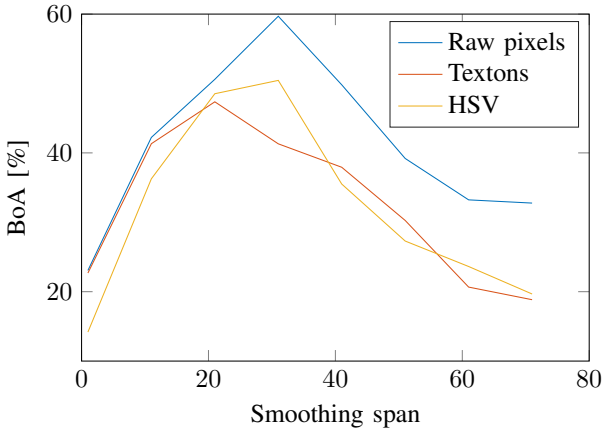


Fig. 12. Different BoAs for varying smoothness of the familiarity curve, evaluated in the photoroom environment. The x-axis indicates the smoothing span used for moving average smoothing. The curves are averages from all locations in the photoroom.

familiarity curves with a span of 30° leads to big increases in performance.

B. Translation

To test familiarity sensitivity with translation only, images taken in a grid pattern are analyzed. In both the sports hall and photoroom the trained view is obtained in the centre of the room. That view is matched against views from the entire room. The heading angle of each view is equal to the heading angle of the trained one. In contrast to rotation, translational motion is not directly controlled. For homing, only the heading angle is adjusted in order to reach the correct destination. This means that good performance in translation is characterized by a familiarity that only changes slightly for small displacements. Stated differently: when a 360° turn is performed, it is advantageous when the familiarity curves are similar for proximate locations, so that good homing performance is achieved even when exploration and homing routes do not perfectly align.

Figures 13 and 14 show the results in the sports hall and photoroom respectively, for raw pixel matching, texton histogram matching and HSV color histogram matching. The colors indicate the familiarity of a certain location, and the arrows show the direction to the steepest familiarity increase in the surroundings.

TABLE II
FAMILIARITY PERFORMANCE METRICS FOR EACH IMAGE MATCHING METHOD AND ENVIRONMENT COMBINATION.

	Raw pixels	Textons	HSV
Sports hall			
Number of local optima	2	5	4
Photo room			
Number of local optima	3	1	2

From the figures it is clear that raw pixel matching shows the most distinct global optimum. Texton and HSV histogram

matching however, show a larger region of optimal familiarity. This can be useful when the robot is slightly off-track, because rotational performance will be similar on adjacent locations. However, in the sports hall environment both methods show several local minima, as shown in Table II, which can be disadvantageous for homing. It is hard to draw conclusions from these minima, since it is not consistent for both environments. Even though raw pixel matching shows the most distinct optimum in the photoroom environment, three local optima are present. It is expected that a larger size of the optimal regions gives more advantages than having a very narrow (distinct) optimum.

VI. Validation Experiment

The environment analyses presented until now are done in simulation. To validate this, an experiment is shown using real imagery taken in an indoor environment. The environment used is the Cyberzoo; a flight arena of the TU Delft. Figure 15 shows two images taken there.

Validation is done for both rotation and translation. For rotation, a total of 25 videos of rotations on the spot are recorded, in a grid of 5 by 5 meters. The average BoAs, peak ratios and correlation coefficients are computed, as done with the simulations presented in the previous section. The results, including the corresponding standard deviations, are shown in Table III. The first observation is that the BoAs are much smaller than in simulation. This is explained by more spikes (and hence local optima) in the results, which is confirmed by the lower correlation coefficients. It is however, in contrast with the observation in the previous section that more realistic environments yield in higher BoAs.

The second observation is that texton and HSV matching show slightly better BoAs than raw pixel matching. Due to the small differences and the large standard deviations however, no significant conclusions can be drawn from this. Still, it may be an indication that texton and HSV histogram matching perform better in real flights. The corresponding rotation plots are shown in Figure 16. From these, it can indeed be seen that the average familiarity curve for texton and HSV histogram matching have a clearer peak at the trained heading.

TABLE III
Familiarity performance metrics for each image matching method in the Cyberzoo environment.

	Raw pixels	Textons	HSV
Rotation			
BoA average	9.13%	12.7%	11.7%
BoA std. dev.	3.38%	6.57%	4.24%
Peak Ratio average	0.52	0.41	0.37
Peak Ratio std. dev.	0.054	0.095	0.093
Correlation Coefficient average	0.82	0.92	0.84
Correlation Coefficient std. dev.	0.093	0.025	0.14
Translation			
Number of local optima	1	2	2

Translation is validated by comparing images taken in the same direction, in a grid of 49 locations. The results are quite similar to the simulation results, and are shown in Figure 17.

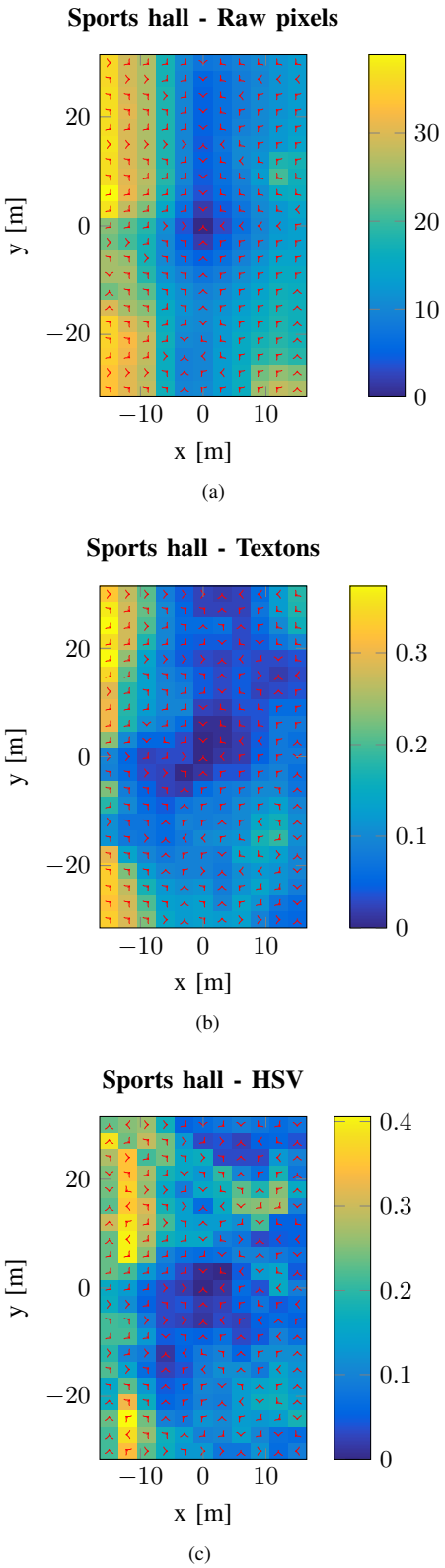


Fig. 13. Varying x and y positions in a SmartUAV simulation, with constant heading angle. Unfiltered images of 48 by 32 pixels are taken in a grid pattern and compared to a stored image at the center of the grid (x=0 and y=0). The environment used is a sports hall, and the camera is pointing upward (in the figure).

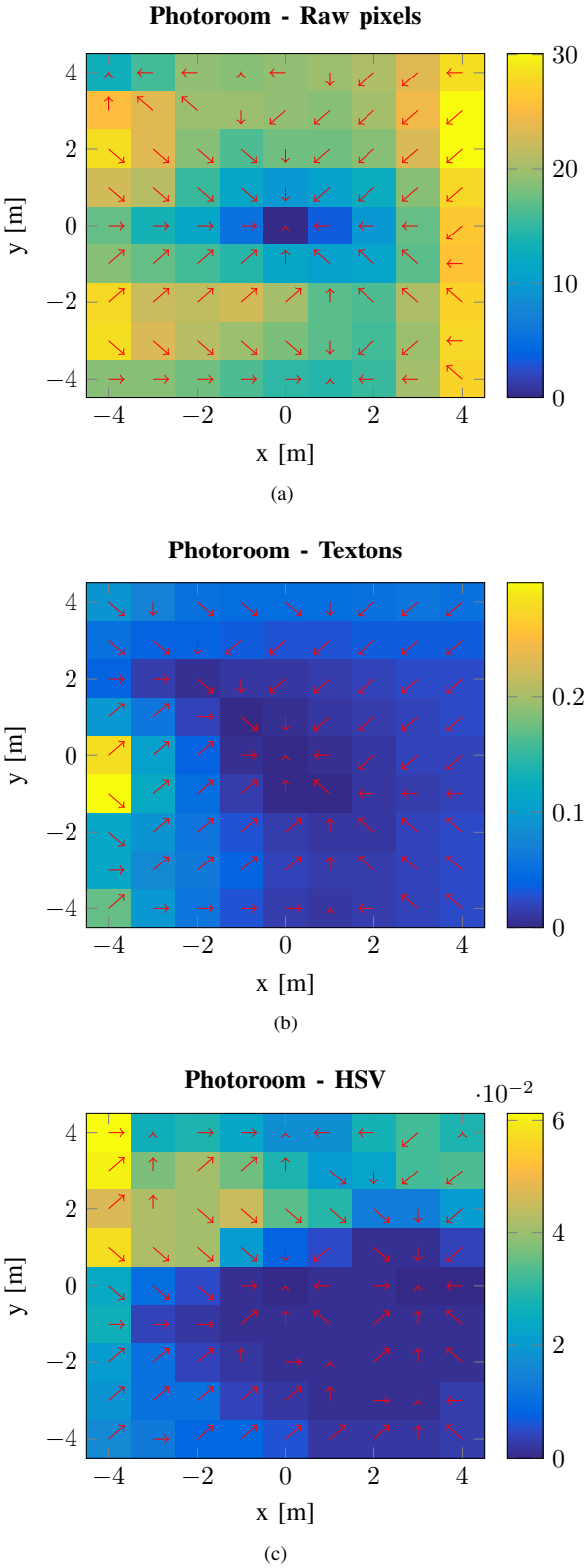
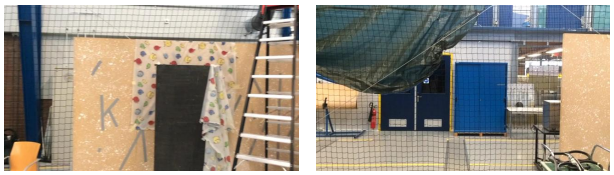


Fig. 14. Varying x and y positions in a SmartUAV simulation, with constant heading angle. Unfiltered images of 48 by 32 pixels are taken in a grid pattern and compared to a stored image at the center of the grid (x=0 and y=0). The environment used is a photoroom., and the camera is pointing to the left (in the figure).



(a) (b)

Fig. 15. Example images from the Cyberzoo environment.

Again, the result for raw pixel matching shows a very narrow peak at the trained location. This can be disadvantageous for homing, since a small offset from the training path can cause divergence from this path. When looking at the texton matching result, it can be seen that two clear optima are present. Even though the surrounding region has quite similar familiarity values, the local optimum at $x = 3m$ and $y = 2m$ might result in wrong convergence.

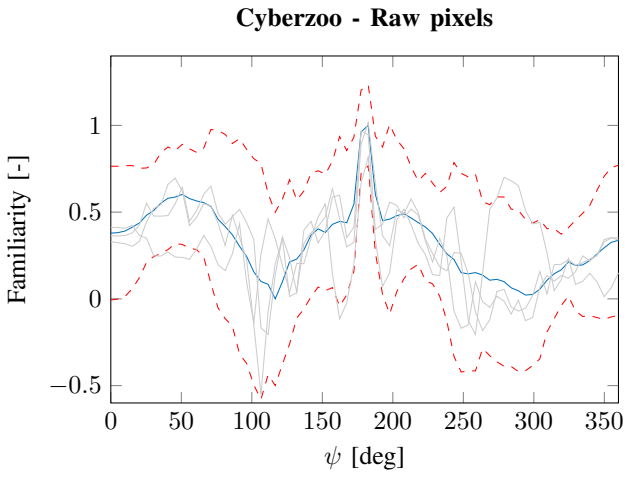
Looking at both rotation and translation of HSV matching, it can be observed that the real-life results are better than those made in simulation. This can be explained by more distinct color in the validation imagery, such that the HSV histogram shows better distribution, enabling more information storage in a single histogram.

VII. Closed-loop Simulation Flight

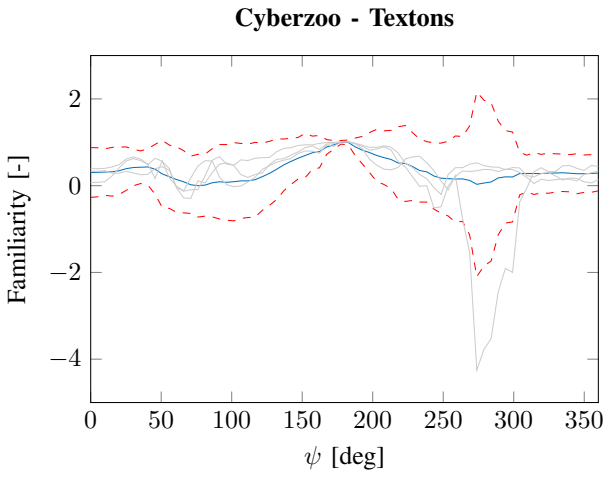
As mentioned in the previous sections, the recognition of views during rotation performs best for texton histogram matching. In simulation, the result is similar to raw pixel matching, but in the validation experiment raw pixel matching performed less well. When observing familiarity during translations, both texton and HSV histogram matching show a large central region of similar familiarity. As explained earlier, this can be advantageous for homing, since recognizing the correct heading during rotation yields a similar familiarity curve for proximate locations. When considering closed-loop simulations, it is therefore expected that texton matching will perform better than the other two methods.

To confirm this, a simulated robot is placed in the sports hall environment. A route is learned by flying backwards (with a speed of $0.5m/s$), such that the front camera looks in the direction of homing, which is necessary to use scene familiarity. One third of the image taken at the center of each view is stored as trained sample. When homing is initiated, the robot starts flying forward with a constant speed of $0.5m/s$ and the heading is constantly determined using view familiarity. This is done by selecting one third of the image giving the best match with one of the trained views. The center of this image patch is converted to an angle, to which the MAV is steered. Views are obtained from a forward looking camera with a field of view of $90°$. This means each stored view has a field of view of $30°$. It also means that it is expected that the method will only work for small turns.

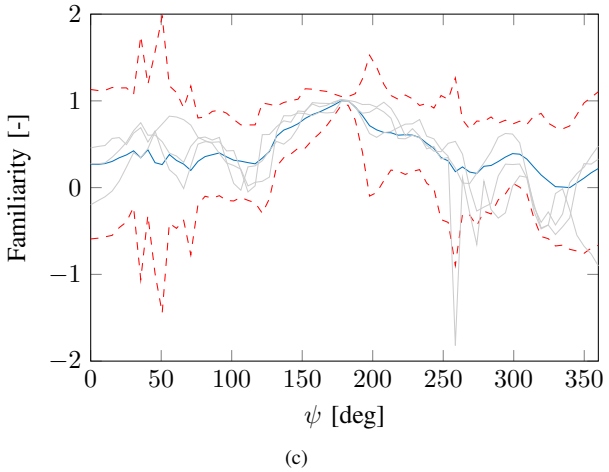
In total, three closed-loop simulations are performed. The first result is shown in Figure 18. The explored route consists of turn angles drawn from a normal distribution with a



(a)

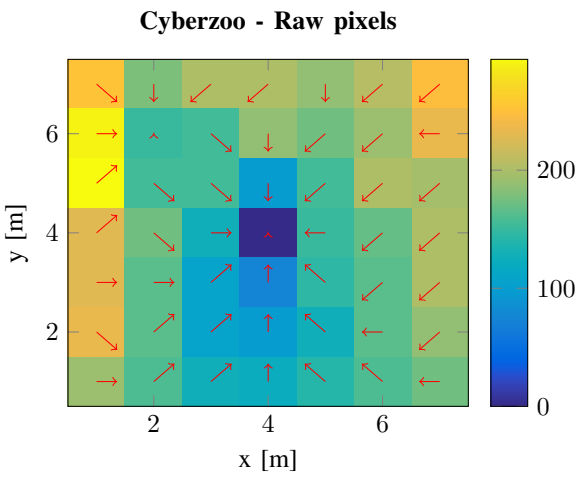


(b)

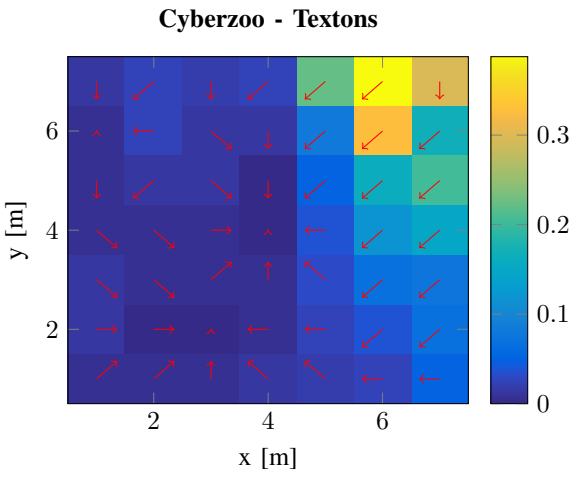


(c)

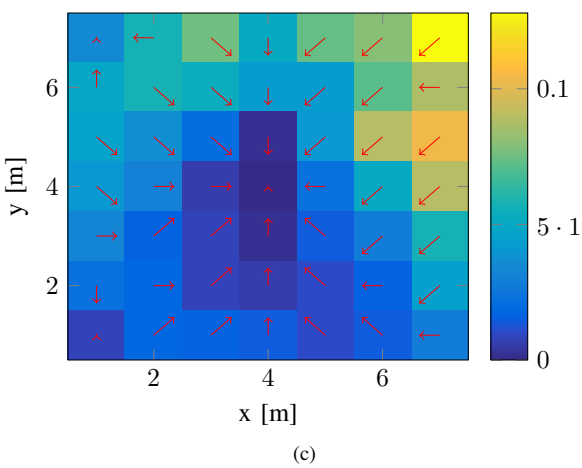
Fig. 16. Average rotation on the spot of 25 locations in the Cyberzoo environment. Unfiltered images of 64 by 36 pixels are taken every 5° and compared to a stored image at a heading angle of 180°. The red dashed lines indicate the 2σ bounds and the gray lines are some example familiarities.



(a)



(b)



(c)

Fig. 17. Varying x and y positions using pictures of the Cyberzoo environment, with constant heading angle. Unfiltered images of 64 by 48 pixels are taken in a grid pattern and compared to a stored image at the center of the grid ($x = 4m$ and $y = 4m$). The camera is pointing upward (in the figure).

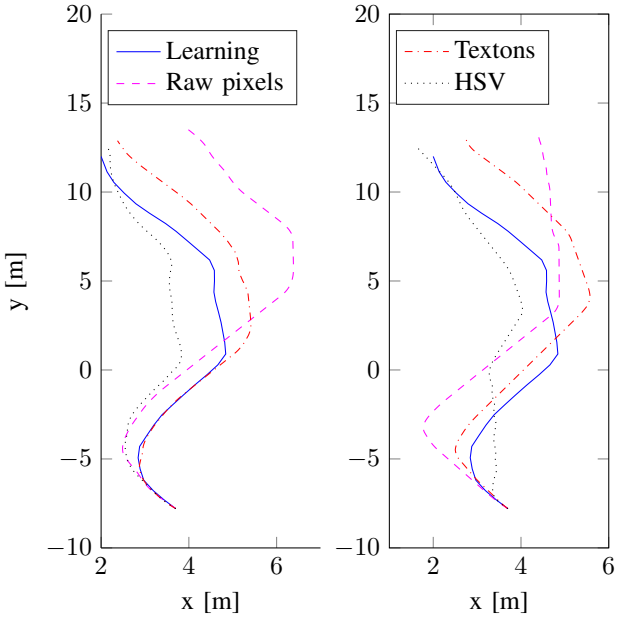


Fig. 18. Closed-loop homing simulation in the sports hall environment in SmartUAV. On the left, a perfect memory is used; on the right the Infomax neural network is applied. The route consists of turns, with an angle drawn from a normal distribution, with a standard deviation of 15°.

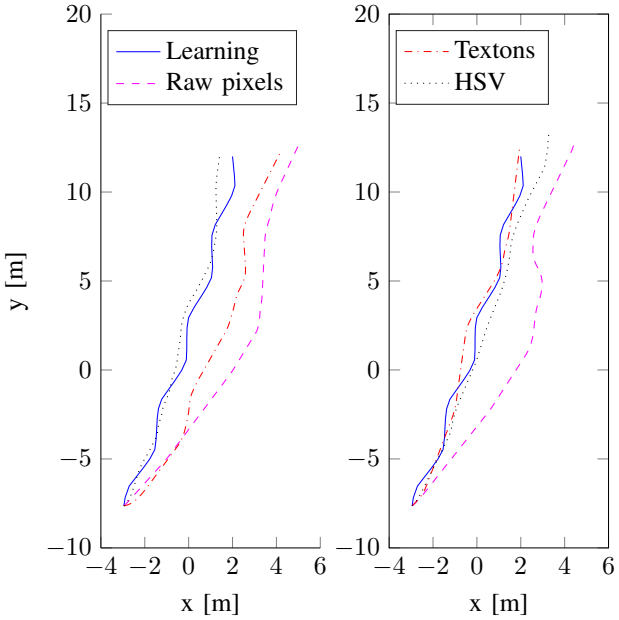


Fig. 19. Closed-loop homing simulation in the sports hall environment in SmartUAV. On the left, a perfect memory is used; on the right the Infomax neural network is applied. The route consists of small, constant turns in alternating direction.

standard deviation of 15°. The left image shows the result for homing using a perfect memory. The blue solid line indicates the trained route, starting at $x = 4m$ and $y = 12m$, which are arbitrarily chosen. A route is flown up to a distance of approximately $20m$ from the starting location.

From the results it can be seen that homing using texton histogram matching or HSV histogram matching approximately reach the initial location. The main difference is that when using texton histogram matching turns are performed with a small delay, where performing homing with HSV histogram matching, turns are made too early. The delay can be explained by a low frequency of the algorithm; because all possible patches are extracted from each image, texton matching operates at approximately $1Hz$, where HSV matching operates at approximately $20Hz$. Texton histogram matching can be significantly improved by using sub-sampling of textons, instead of extracting them all. The path taken when performing homing using HSV histogram matching is cut off, although the curvature of the trained path is followed. When homing is done by matching raw pixels (performed at approximately $5Hz$), the robot diverges from the trained route. It does, however, also follow the curvature of the trained path. The fact that raw pixel matching performs less suggests that differences in familiarity when a vehicle drifts causes views to be hard to distinguish.

The Infomax neural network can be used as function approximator of familiarity [27]. To test this in closed-loop, the three methods are all represented in a neural network. For both texton and HSV histogram matching a network with 50 inputs is defined (i.e., each histogram forms one input vector to the network). The number of novelty neurons is chosen to be 200. Furthermore, the number of epochs is set to 500. It turned out that a lower number of epochs gives significantly less performance. In Future simulations or flight tests this should be tuned by testing multiple amounts of both novelty neurons and epochs. For raw pixel matching, the image is scaled down to 16 by 12 pixels, which gives 192 inputs to the network. Larger dimensions as input cannot be processed in real-time. The number of novelty neurons and epochs are kept the same.

The results of homing along the same route using an Infomax network can be seen at the right of Figure 18. It is clear that the results are slightly worse than with a perfect memory (i.e., by keeping a database of images, textons or HSV histograms). It does however, look quite similar to the perfect memory case, which suggests that the assumption that Infomax is a valid approximator for familiarity is quite good.

To analyze more diverse exploration routes, two additional closed-loop simulations are performed. These results are shown in Figures 19 and 20. The result in Figure 19 performs similar to the previous one. Again, raw pixel matching performs less well than the other two methods. HSV histogram matching however, performs slightly better than texton histogram matching. Another notable observation is that the Infomax result of texton histogram matching performs better than the perfect memory one. Even though this is odd, it can be caused by the fact that evaluation of the Infomax neural network is very fast (which does not hold for training it though). Again, it is questionable whether Infomax still works bettter when sub-sampling of textons is applied.

The final closed-loop simulation results are shown in Fig-

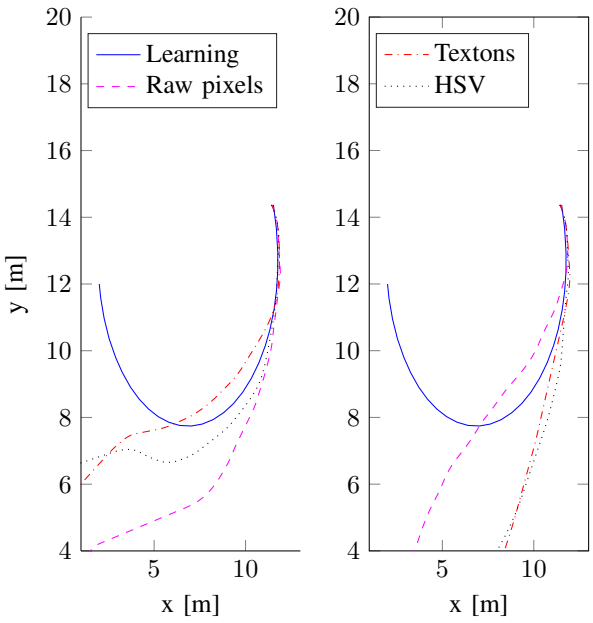


Fig. 20. Closed-loop homing simulation in the sports hall environment in SmartUAV. On the left, a perfect memory is used; on the right the Infomax neural network is applied. The route resembles half a circle.

ure 20. Here, half a circle is flown during training, to test how well the method performs with big turns. It can be seen that neither of the methods reach the initial location, although they all get halfway. Unfortunately, no definite conclusions can be drawn from this, because the algorithm is implemented for small turns only. Especially for raw pixel matching and HSV histogram matching, the divergence from the route explains the fact that the starting location is not reached. Since the turn is not tight enough (again, probably due to the inherently small turns), at some point the trained view cannot be seen anymore. When performing homing using texton histogram matching, the turn is made a little too tight. When the exploration path is crossed, the right turn needed to get back on track is too sharp. Again, the last result aims to give an impression of turn performance only. From this it cannot be concluded that scene familiarity based algorithms cannot deal with larger turns.

VIII. Discussion

When first looking at the rotational analysis, it was observed that raw pixel and texton histogram matching performed best. When looking at the translation results, raw pixel matching shows the most distinct peak. Because position of the robot is not directly controlled, it is advantageous that a large familiar region appears in translation, so that a little drift of the robot does not change the homing performance. This was especially the case for texton histogram matching and HSV histogram matching. This suggests that texton histogram matching would perform best, which is confirmed by the closed-loop results. Even though the final error from the home location was sometimes smaller with HSV histogram matching, the overall tracking of the route was better with texton histogram matching.

Surprisingly, HSV histogram matching shows very good performance in closed loop. A reason for this can be that generating and storing HSV histograms is computationally very efficient, which allows for a high frequency of the algorithm. This means corrections are made very quickly so that the robot does not diverge much. Especially because the algorithm can only perform small turns due to the limited field of view, running at a high frequency enables a higher *turn rate*. It does not say however, that HSV matching would perform well when an agent has already diverged from the route.

When evaluating the closed-loop tests in this paper, some limitations can be identified. First of all, it is only tested in simulation. Even though the fidelity of the simulation is higher than the simulations performed by Baddeley et al., it is questionable whether the same results would be obtained in a real flight. Furthermore, many additions can be proposed to make the algorithm more robust. An example is to use active rotation instead of using the inherent field of view of the forward looking camera, so that bigger turns can be made. Another possible addition is the use of visual odometry to get a rough estimate of the path taken. Odometry could be used to prevent severe divergence from the correct route. This was especially seen in the last closed-loop result, where half a circle was flown during exploration. Visual odometry can lead to approximate following of the route and the integration errors caused by visual odometry could be solved by using scene familiarity. Since the experiment enforces the implementation of small turns only, it cannot be concluded that the method works well for diverse trajectories, which is, again, seen in the last closed-loop result.

Another point of discussion is that the main reason scene familiarity can be a viable approach for visual homing of MAVs is computational efficiency. The only way this is tested in this paper, is by performing the closed-loop simulations in real-time on a laptop computer. When implementing the algorithm on-board an MAV, the real-time performance may be inadequate due to a slower micro-processor. This seems less of a problem for HSV histogram matching, because both the computations needed to extract histograms and the memory requirements are limited. In this paper however, all textons were extracted from each image. Usually, it suffices to randomly pick a set of textons (i.e., sub-sampling), which would drastically improve computational performance. The storage of a texton histogram is similar to storing an HSV histogram.

As explained in section III Baddeley et al. use an Infomax neural network for familiarity representation. Even though results with the network are shown in the closed-loop tests, this cannot be considered as an analysis based on which conclusions can be drawn about this network. In this paper, the assumption is made that using a database of views (in Baddeley et al. referred to as a *perfect memory*) always gives better performance than a neural network, given that computational efficiency is disregarded. In the closed-loop

results it is seen that disregarding the computational efficiency should not be done, because Infomax performed better than using a perfect memory once. In general, the performance when using an Infomax neural network was quite similar to the perfect memory experiments. For this reason, using such a neural network should be considered, because the memory usage can be controlled and limited. Although this means that the network can *forget* earlier trained views, it allows control over the often very limited memory available on MAVs. Another advantage is that evaluating familiarity can be done very fast. Training on the other hand, is quite slow; especially when having to train each sample 500 times. When the method is combined with a path integration method, this can be solved by increasing the timestep between different training instances.

A final note about the simulations presented in this paper, is that the image representations are not tuned in a structured manner. Examples of tunable parameters are image dimensions, number of texton clusters, the texton clustering method, the number of patches extracted for histogram generation and the HSV saturation threshold. Tuning this, might either increase performance or decrease computational cost, which are both advantageous.

IX. Conclusion and Recommendations

This paper investigates the applicability of the scene familiarity homing method, observed from insect behavior, to MAVs. The scene familiarity method is introduced as proof of concept for desert ants to use recognition of a route to find their way home. Next to this, an unsupervised neural network method was used to limit the memory required for storing familiarity.

The concept of only using recognition along a route is a very interesting one. The analysis shows that the closed-loop performance is good; at least for straight, short paths. The reason the method is promising, is the potential computational efficiency. For all three image representations the algorithm works in real-time on a laptop computer, but for texton histogram matching and raw pixel matching the frequency is quite low. As mentioned, extracting all textons from an image is computationally expensive and probably not necessary. It is therefore recommended that experiments are done with sub-sampled image patches. We expect that the increase in efficiency inherently increases the performance, due to the higher possible turn rate. This combined with the currently good tracking of texton histogram matching, brings us to the conclusion that texton histograms are the most promising image representation for visual homing using scene familiarity.

In the closed-loop results, it is seen that once a route is lost, the current implementation is not robust enough to find it back. As mentioned, combining scene familiarity with visual odometry can improve this. Visual odometry can further be used for stopping at the home location. Another solution is to change the algorithm such that the vehicle actively looks around to observe the scene, instead of using the field of view of the forward looking camera. This should be investigated in future research, as well as closed-loop experiments done on real-life MAVs.

The most surprising results came from HSV histogram matching (either with or without the Infomax neural network), since performance is good and it is very fast. This makes it a good method to be combined with other, existing methods like visual odometry. This seems especially needed because the environment analysis showed that HSV histogram matching results were quite noisy.

The final conclusion is that when computational efficiency is a driving factor, scene familiarity is a viable approach for visual homing. Using an Infomax neural network is advantageous, because the results showed similar performance, while being more efficient during homing. Especially when complementing the scene familiarity method with visual odometry, the method can be a powerful alternative to existing visual navigation solutions.

References

[1] Z. Mathews, M. Lechon, J. B. Calvo, A. Dhir, A. Duff, S. B. i Badia, and P. F. Verschure, "Insect-like mapless navigation based on head direction cells and contextual learning using chemo-visual sensors," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. New York, USA: IEEE, oct 2009. 1

[2] M. V. Srinivasan, "Where paths meet and cross: navigation by path integration in the desert ant and the honeybee," *J Comp Physiol A*, vol. 201, no. 6, pp. 533–546, may 2015. 1

[3] R. C. Nelson, "Visual homing using an associative memory," *Biological Cybernetics*, vol. 65, no. 4, pp. 281–291, aug 1991. 1

[4] B. Baddeley, P. Graham, P. Husbands, and A. Philippides, "A model of ant route navigation driven by scene familiarity," *PLoS Computational Biology*, vol. 8, no. 1, jan 2012. 1, 3, 4, 5, 6

[5] C. R. Gallistel, *The Organization of Learning*. Cambridge, MA, USA: MIT Press, 1990. 1

[6] J. Leonard and H. Durrant-Whyte, "Simultaneous map building and localization for an autonomous mobile robot," in *Proceedings IROS '91:IEEE/RSJ International Workshop on Intelligent Robots and Systems '91*. New York, USA: IEEE, 1991. 2

[7] F. Bonin-Font, A. Ortiz, and G. Oliver, "Visual navigation for mobile robots: A survey," *Journal of Intelligent Robotic Systems*, vol. 53, no. 3, pp. 263–296, may 2008. 2

[8] K. L. Ho and P. Newman, "Detecting loop closure with scene sequences," *International Journal of Computer Vision*, vol. 74, no. 3, pp. 261–286, jan 2007. 2

[9] L. Clemente, A. Davison, I. Reid, J. Neira, and J. Tards, "Mapping large loops with a single hand-held camera," in *Proceedings of Robotics: Science and Systems*. Atlanta, GA, USA: MIT Press, June 2007. 2

[10] B. P. Williams, "Simultaneous localisation and mapping using a single camera," Ph.D. dissertation, Oxford University, 2009. 2

[11] E. Motard, B. Raducanu, V. Cadenat, and J. Vitria, "Incremental on-line topological map learning for a visual homing application," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. New York, USA: IEEE, apr 2007. 2

[12] G. Desouza and A. Kak, "Vision for mobile robot navigation: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 237–267, 2002. 2

[13] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques," *International Journal of Computer Vision*, vol. 12, no. 1, pp. 43–77, feb 1994. 2

[14] H. Chao, Y. Gu, and M. Napolitano, "A survey of optical flow techniques for robotics navigation applications," *Journal of Intelligent & Robotic Systems*, vol. 73, no. 1-4, pp. 361–372, oct 2013. 2

[15] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, ser. IJCAI'81. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1981, pp. 674–679. 2

[16] B. K. Horn and B. G. Schunck, "Determining optical flow," in *Techniques and Applications of Image Understanding*, J. J. Pearson, Ed. SPIE, nov 1981. 2

[17] M. V. Srinivasan, "An image-interpolation technique for the computation of optic flow and egomotion," *Biological Cybernetics*, vol. 71, no. 5, pp. 401–415, sep 1994. 2

[18] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. New York, USA: IEEE, 2004. 2

[19] B. A. Cartwright and T. S. Collett, "Landmark learning in bees," *Journal of Comparative Physiology*, vol. 151, no. 4, pp. 521–543, 1983. 2

[20] ——, "Landmark maps for honeybees," *Biological Cybernetics*, vol. 57, no. 1-2, pp. 85–93, aug 1987. 2

[21] M. O. Franz, B. Schölkopf, H. A. Mallot, and H. H. Bülthoff, "Where did i take that snapshot? scene-based homing by image matching," *Biological Cybernetics*, vol. 79, no. 3, pp. 191–202, oct 1998. 2

[22] T. S. Collett, "Insect navigation en route to the goal: Multiple strategies for the use of landmarks," *Journal of Experimental Biology*, vol. 199, pp. 227–235, 1996. 2

[23] R. Möller, M. Maris, and D. Lambrinos, "A neural model of landmark navigation in insects," *Neurocomputing*, vol. 26-27, pp. 801–808, jun 1999. 2

[24] D. Lambrinos, R. Möller, T. Labhart, R. Pfeifer, and R. Wehner, "A mobile robot employing insect strategies for navigation," *Robotics and Autonomous Systems*, vol. 30, no. 1-2, pp. 39–64, jan 2000. 2

[25] D. Lambrinos, R. Möller, R. Pfeifer, and R. Wehner, "Landmark navigation without snapshots: the average landmark vector model," in *Proceedings of Neurobiology Conference Göttingen*, 1998. 2

[26] A. Cheung, M. Collett, T. S. Collett, A. Dewar, F. Dyer, P. Graham, M. Mangan, A. Narendra, A. Philippides, W. Stürzl, B. Webb, A. Wystrach, and J. Zeil, "Still no convincing evidence for cognitive map use by honeybees," *Proceedings of the National Academy of Sciences*, vol. 111, no. 42, pp. E4396–E4397, oct 2014. 3

[27] A. Lulham, R. Bogacz, S. Vogt, and M. W. Brown, "An infomax algorithm can perform both familiarity discrimination and feature extraction in a single network," *Neural Computation*, vol. 23, no. 4, pp. 909–926, apr 2011. 3, 4, 14

[28] J. Zeil, M. I. Hofmann, and J. S. Chahl, "Catchment areas of panoramic snapshots in outdoor scenes," *Journal of the Optical Society of America A*, vol. 20, no. 3, p. 450, mar 2003. 3, 6

[29] A. J. Bell and T. J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, no. 6, pp. 1129–1159, nov 1995. 4

[30] T.-W. Lee, M. Girolami, and T. J. Sejnowski, "Independent component analysis using an extended infomax algorithm for mixed subgaussian and supergaussian sources," *Neural Computation*, vol. 11, no. 2, pp. 417–441, feb 1999. 5

[31] D. D. Gaffin and B. P. Brayfield, "Autonomous visual navigation of an indoor environment using a parsimonious, insect inspired familiarity algorithm," *PLOS ONE*, vol. 11, no. 4, p. e0153706, apr 2016. 5

[32] M. Varma and A. Zisserman, "A statistical approach to texture classification from single images," *Int J Comput Vision*, vol. 62, no. 1-2, pp. 61–81, apr 2005. 6

Chapter 3

IMAV Article

This chapter contains the conference paper submitted to IMAV 2016, taking place in Beijing, China. It is a more consize article than the main one in the previous chapter and focusses on the same research and results.

Visual Homing for Micro Aerial Vehicles using Scene Familiarity

Gerald J.J. van Dalen,* Kimberly N. McGuire, and Guido C.H.E. de Croon
Delft University of Technology, The Netherlands

ABSTRACT

Autonomous navigation is a major challenge in the development of Micro Aerial Vehicles (MAVs). Especially when an algorithm has to be efficient, insect intelligence can be a source of inspiration. An elementary navigation task is homing, which means autonomously returning to the initial location. A promising approach uses learned visual familiarity of a route to determine reference headings during homing. In this paper an existing biological proof-of-concept is transferred to an algorithm for micro drones, using vision-in-the-loop experiments in indoor environments. An artificial neural network determines which control actions to take.

1 INTRODUCTION

A major challenge in robotics is to navigate autonomously through an unknown environment. Especially in indoor scenes, where no Global Positioning System (GPS) system is available, the entire navigation problem is not yet solved.

Current navigation algorithms either require expensive sensors or significant computation power. Especially Simultaneous Localization and Mapping (SLAM) methods have shown to be successful in real-time navigation, given enough computational power on-board a vehicle or good sensors. Most Micro Aerial Vehicles (MAVs) do not have such sensors and cannot perform heavy computations on-board the vehicle.

In order to find suitable navigation algorithms for MAV, insects can be a source of inspiration, since they constantly have to deal with complex navigation problems while only having small-sized brains [1]. Different algorithms have already been created based on observations done on insects. A well-known example is using optic flow to get a sense of velocity, which is known to be done by insects [2]. Integrating this estimate for localization is called visual odometry. The obtained location estimate is employed in higher level navigation algorithms. Still, these algorithms are not readily available for tiny MAVs yet. One of the higher level skills employed by insects is the ability to return to the nest location. This is referred to as *homing* [3]. It would be an important enabler for MAVs, if they could use similarly high-level, but computationally efficient algorithms for navigation.

*Email address: gjj.vandalen@gmail.com

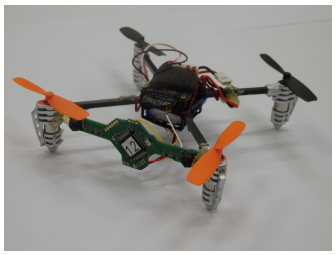


Figure 1: Pocket drone: a micro quad rotor containing a Lisa-S autopilot and a stereo camera [5]. While this pocket drone can already fly, stabilize and avoid obstacles, in this paper we investigate efficient insect-inspired algorithms that will allow it to navigate in an unknown environment.

A promising homing algorithm is proposed by Baddeley et al., where familiar views along a route are used to determine the correct direction to an earlier visited location [4]. This is a *visual* homing algorithm, since cameras are used as driving sensor. Instead of focussing on the contruction of a detailed (or coarse) map, Baddeley et al. propose that homing can be performed just by means of recognizing which direction seems most familiar to a robot. Furthermore, they use a small neural network to store and recapitulate a route in order to find the initial location. Potentially, this is very useful for MAV navigation algorithms, since it deals with limited storage capacity found on many small platforms, like the pocket drone shown in Figure 1.

In an effort to find efficient navigation algorithms for MAVs, this paper investigates the practical application of the scene familiarity algorithm on MAVs. The focus is on how robust familiarity is to determine control actions.

First, section 2 discusses the state-of-the-art in autonomous visual navigation on drones. Then, section 3 explains the scene familiarity method as introduced by Baddeley et al. Section 4 shows simulations and experiments for different environments, to overcome current shortcomings in the implementation described by Baddeley et al. Finally, closed-loop simulation flights are performed and presented in section 5, to show a more realistic use-case of view familiarity for MAV homing.

2 RELATED RESEARCH

This section gives a brief overview of previous research done to visual navigation and specifically visual homing. Visual SLAM is the most commonly used algorithm in camera-

driven robotics. An example is shown in Motard et al., where an AIBO robot[1] must navigate back to its charging station [6]. Still, visual SLAM algorithms in real-time require much computational resources, since (visual) processing, mapping and self-localization must be performed simultaneously. Since most MAVs have limited computational resources, visual SLAM often cannot be run in real-time, which makes it less suitable for homing.

In 1983, Cartwright & Collett introduced the Snapshot Model [7]. The framework they presented gives an explanation of the navigation capabilities of bees when traveling between different food sources. The visual matching is done by a direct comparison of an image on the retina with a stored snapshot. The landmark approach is further extended by the addition of visual beacons [8]. A disadvantage of this is that many images have to be stored.

A similar approach uses Average Landmark Vectors (ALVs) to represent landmarks [9]. ALVs, introduced by Lambrinos et al. in 1998, are averages of the heading vectors to all landmark locations [10]. The homing vector is determined with respect to this ALV. ALV homing stores the location of interest as a vector, which is more efficient in computation and storage, than storing an entire image. However, due to its simplicity, ALV homing is also more prone to errors.

Scene familiarity methods refer to recognition of a traversed route, without specific information about the goal location. This means, a robot must always move into the *most familiar* direction. In the ideal case, this would automatically mean that the agent returns to the goal location. In 2012, a scene familiarity method is proposed for visual homing of desert ants [4]. The scene familiarity method proposed by Baddeley et al. is quite new and not yet used in robotic applications. The next section reviews their paper in depth.

3 The Scene Familiarity Method

In an effort to find a biologically more plausible alternative to map-based navigation methods and the snapshot model described in the previous section, the scene familiarity homing method is introduced [4]. To show that homing navigation could take place without the use of visual odometry, a method is presented where views along the entire route determine the heading in which to proceed. Conceptually, this means that during a training run images in the direction of the route are stored. Then, when using the algorithm for homing, images taken around the robot are compared to these stored views in order to determine the most familiar direction.

When the homing capabilities are tested, the agent is placed back at its initial location. From there, homing is done by performing 360° scans of the world and comparing images taken in each direction with all images stored. A familiarity value of a single image is obtained by calculating the Sum of Squared Differences (SSD) of raw pixel values, as defined in Equation 1 [11].

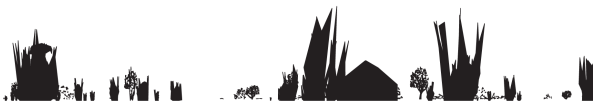


Figure 2: Binary panoramic image used in Baddeley et al. [4].

$$F(I) = -\arg\min_i \sum_{x,y} \left(I(x,y) - V_i(x,y)\right)^2 \qquad (1)$$

In this equation, $F(I)$ indicates the familiarity of view I, $I(x,y)$ is the current view and $V_i(x,y)$ are the stored views. It can be seen that the stored image that gives the closest match to the current image is used as familiarity value. The agent can rotate on the spot or use an omni-directional camera to obtain familiarity values in all directions. After determining the most familiar direction (by maximizing the values obtained with Equation 1), the simulated agent is moved in that direction.

The stored panoramas are binary images and have dimensions of 90 by 17 pixels (Figure 2). The resolution is such that each pixel in horizontal direction is equivalent to a rotation of 4°. During homing, familiarity is evaluated for steps of 1 pixel, such that Equation 1 is evaluated 90 times. The maximum outcome of this results in the most familiar direction.

Due to the large memory needed for storing images and the computational requirements, the algorithm in the current form is not yet suitable for implementation on-board a small robot. Baddeley et al. therefore also study an unsupervised Infomax neural network to approximate familiarity [12]. The network is a two-layer neural network, where the linear combination of an input and the network weights represent familiarity. A lower value indicates *more familiar*. The training rule therefore adapts the weights such that the value is lower for every input encountered during training.

Baddeley et al. showed the validity of scene familarity with virtual robotic ants in a simulated environments. However, they use an environment of binary sceneries, which are not representative for the scenes through which a robot must navigate. Moreover, the simulation is set up such that moving the image by one pixel in the horizontal direction is equivalent to a rotation of the agent of 4°. These direct relations to rotation and pixel difference are not realistic for real-life cameras. Furthermore, the algorithm has only been tested on relatively small distances, since images are stored every $4cm$ and movements of $10cm$ per timestep are made. When the method is implemented in robotics, the robot should be able to cover longer distances to make it more useful.

Recently, Gaffin et al. have published a detailed analysis on scene familiarity in realistic, indoor environments [13].

[1] http://www.sony-aibo.co.uk/

Distinguishing familiarity is both analyzed in rotation and translation, for raw pixel matching between images of different resolutions. A rail mounted camera is used to perform a MATLAB-driven experiment.

In our analysis of the scene familiarity method, we will use a simulator containing realistic sceneries, vehicle dynamics and camera parameters. A translation and rotation analysis will be performed as well, however, next to raw pixel values, we will also investigate alternative image representations, to determine which one is more suitable for recognizing familiar views. Closed-loop simulations with an MAV are presented and we show the use of an Infomax neural network as well, since this helps in meeting the limited storage requirements of an MAV. We hope to better understand autonomous navigation for small MAVs.

4 Familiarity Analysis

In the previous section, the original simulation results presented by Baddeley et al. are discussed [4]. Based on this, a key question remains whether the algorithm will work in more realistic environments. In this and the following sections, an analysis of an indoor simulated environment is presented in combination with different image representation methods. First, the tested image representations and calculated performance measures are introduced. Then, simulation results of these different methods in multiple environments are shown. To validate this, similar results are shown on real imagery.

4.1 Methods

To test the usability of familiarity of scenes for visual homing, we investigate the familiarity sensitivity during both rotation and translation. Analyzing rotation is done by performing a 360° turn at a fixed location in the environment, in steps of 5°. A single image is stored and used as trained view and all other views experienced during this rotation are compared to this. The hypothesis is that familiarity should improve when the heading difference between the current view and the stored image decreases.

Translation is analyzed by evaluating familiarity in a grid of locations, with a fixed heading. Again, a single image is used as training sample and the familiarity is expected to improve when the distance to the trained view gets smaller. Results of this should show the sensitivity of familiarity with both increasing distance (in two directions) and increasing heading angle.

The following image representations are compared:

- **Raw pixel values** The sum of squared differences of each pixel in two images outputs a similarity score [11], as shown in Equation 1.
- **Texton histograms** Textons are small distinct image patches, which can be extracted from an image [14]. When clustered with a texton dictionary, histograms are formed which represent an image.

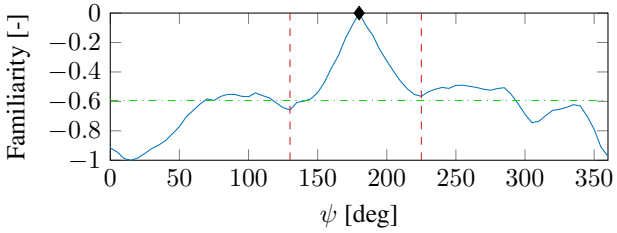


Figure 3: Rotation on the spot at a constant location in a simulator. Unfiltered images of 48 by 32 pixels are taken every 5° and compared to a stored image at a heading angle of 180°. The red dashed lines indicate the BoA bounds and the green dashed line shows the mean familiarity.

- **Hue Saturation Value (HSV) color histograms** Color histograms contain a classification of each pixel based on color intensity.

The performances of the different methods are evaluated by 1) looking at how distinct a view close to the trained view is, compared to other views and 2) what the probability is that the correct (i.e., trained) view is selected as most familiar, since that direction will be chosen for homing. Figure 3 shows an example of a familiarity evaluation when rotating on the spot. The trained image is positioned at an angle of 180° and, in this example, image matching is done using the SSD of raw pixel values. The performance is evaluated using the following measures:

- **Peak ratio** The peak ratio is defines as:

$$PR = \frac{\max F - \mu_F}{\max F - \min F} \tag{2}$$

  In this equation, F refers to the familiarity values shown in Figure 3 and μ_F is the mean of all familiarity values (i.e., the green line in the figure). The higher the peak ratio is, the more distinct a peak is.
- **Basin of Attraction (BoA)** The basin of attraction shows how far an agent can be off from the trained view, before diverging from the correct direction. It is evaluated by finding all local optima (both minima and maxima) and looking between which minima the agent converges towards the trained optimum familiarity (maximum).
- **Correlation coefficient** This is used to estimate the correlation between two neighboring heading angles, differing by 5°. Here, the Pearson product-moment correlation coefficient is used, where 1 indicates full positive correlation between two neighboring angles, -1 means full negative correlation and 0 means no correlation.

The BoA is considered to be most important, since it determines how far an agent can be off the route (i.e., the correct heading), while still being able to converge back to the correct path with a gradient-like search. The peak ratio is mainly

(a) (b)

Figure 4: Examples from the scenery used in SmartUAV simulations (a) and the validation Cyberzoo environment (b).

useful when an agent has no clue where to go; if the agent makes a 360° turn and the trained peak is very distinct, the probability of continuing in the right direction is high. The correlation coefficient gives a measure for how continuous a familiarity curve is. When the correlation is low, it could happen that spikes occur in the familiarity curve, which may give wrong results.

4.2 SmartUAV Simulations

This section shows analyses for sceneries in the SmartUAV simulator. SmartUAV is made for Guidance Navigation & Control (GNC) research on MAVs and specializes in the use of vision as primary sensor. The simulator is written in C++ and sensors and controllers can be connected using a block interface. This makes it easily extendable and the level of simulation fidelity can be adapted by changing complexity of vehicle dynamics, sensor dynamics and realism of the environment.

The tested environment is based on a sports hall located in Delft (the Netherlands). The dimensions are 30 by 60 meters. Figure 4a shows an example view of the sports hall. This environment is used for both familiarity analysis and closed-loop simulations.

As mentioned, both rotational and translational familiarity sensitivity will be tested. For familiarity estimation, SSD values of raw pixels, SSD values of texton histograms and SSD values of HSV color histograms are used and compared. The familiarity sensitivity to yaw rotations is most important for view familiarity-based homing. Each turn taken during homing is made based on the familiarity values for different heading angles. To analyze familiarity for different headings, different image representations are compared by calculating the BoAs, peak ratios and correlation coefficients. An MAV is simulated at a single location and stores a representation of one view. This view is matched to images in all other directions to get a measure of familiarity. This is done in a grid of locations in the sports hall, to get imagery in the center of the room, as well as close to walls. For each location, the BoA, peak ratio and correlation coefficient can be calculated.

Table 1 summarizes these performance measures for the different methods. The calculated BoAs, peak ratios and correlation coefficients are averaged for all locations and the standard deviations are included as well. Good performance is characterized by large BoAs (i.e., it is likely that the correct heading is found), large peak ratios (i.e., the correct familiarity value is distinct compared to familiarities in other directions) and correlation coefficients close to 1 (i.e., continuous and not too noisy).

Table 1: Average performance metrics during rotation, for each image matching method in the simulated sports hall.

	Raw pixels	**Textons**	**HSV**
BoA average	37.3%	36.7%	6.90%
BoA std. dev.	16.5%	12.0%	3.77%
Peak ratio average	0.57	0.43	0.53
Peak ratio std. dev.	0.10	0.076	0.13
Corr. coeff. average	0.98	0.98	0.80
Corr. coeff. std. dev.	0.051	0.0091	0.14

Looking at the results, it can be seen that the BoAs for raw pixel matching and texton histogram matching perform similarly. HSV histogram matching performs much worse, which is also seen in the lower correlation coefficient. This indicates more local optima, which inherently decreases the BoA. The peak ratio is best with raw pixel matching, although the differences between the different methods are quite small.

To illustrate the results shown in the table, familiarity curves are shown in Figure 5. The top plot shows raw pixel matching, the middle texton histogram matching and the bottom one HSV histogram matching. The blue, solid lines indicate the average familiarity curves for all locations in the environment, the red dashed lines indicate two times the standard deviation and the gray lines show some example familiarity curves at individual locations in the sports hall. The results are scaled such that the average lies between 0 and 1.

As expected, all average curves show a single peak at the trained locations (i.e., at 180°). The HSV histogram result however, shows a less predictable outcome, with a larger amount of local optima. This is in line with the lower BoAs and correlation coefficients shown in Table 1.

To test familiarity sensitivity with translation only, images taken in a grid pattern are analyzed. In the sports hall the trained view is obtained in the centre of the room, which is matched against views from the entire room, keeping the heading angle constant. In contrast to rotation, translational motion is not directly controlled. For homing, only the heading angle is adjusted in order to reach the correct destination. This means that good performance in translation is characterized by a familiarity that does not change too much for small displacements. Stated differently: when a 360° turn is performed, it is advantageous when the familiarity curves are similar for proximate locations, so that good homing performance is achieved even when exploration and homing routes do not perfectly align. Figure 6 shows the results in the sports hall environment, for raw pixel matching, texton histogram matching and HSV color histogram matching. The colors

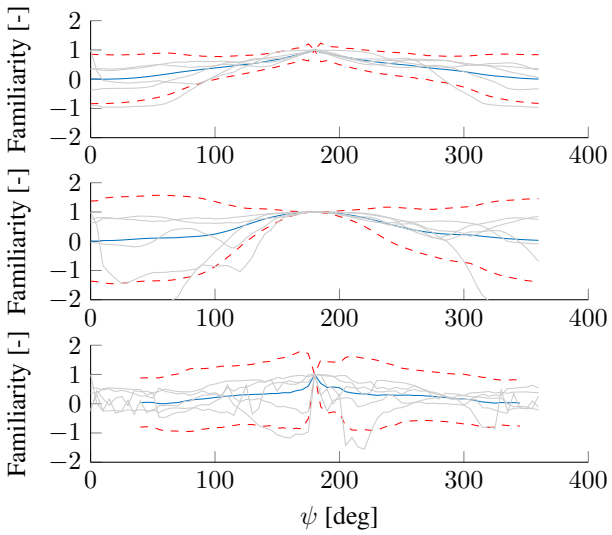


Figure 5: Average rotation on the spot of 231 locations in the sports hall environment in SmartUAV. Unfiltered images of 48 by 32 pixels are taken every 5° and compared to a stored image at a heading angle of 180°. The red dashed lines indicate the 2σ bounds and the gray lines are some example familiarities. The top, middle and bottom plots indicate raw pixel matching, texton histogram matching and HSV histogram matching respectively.

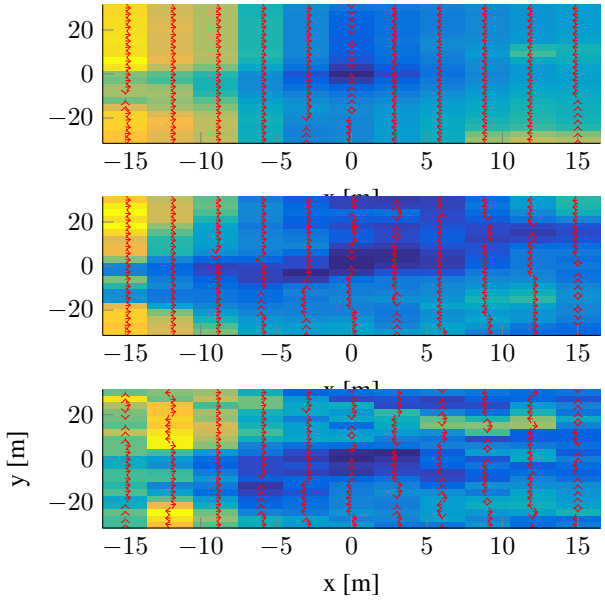


Figure 6: Varying x and y positions in a SmartUAV simulation in a sports hall, with constant heading angle. Unfiltered images of 48 by 32 pixels are taken in a grid pattern and compared to a stored image at the center of the grid (x=0 and y=0). The top figure uses raw pixel matching, the middle figure texton histograms and the bottom figure HSV histograms.

indicate the familiarity of a certain location and the arrows show the direction to the steepest familiarity increase in the surroundings.

From the figures it is clear that raw pixel matching shows the most distinct global optimum. Texton and HSV histogram matching however, show a larger region of optimal familiarity. This can be useful when the robot is slightly off-track, because rotational performance will be similar on different locations. However, both methods show several local minima, which can be disadvantageous for homing.

4.3 Validation Experiment

The previous analysis is done in simulation. To validate this, an experiment is shown using real imagery taken in an indoor environment. The environment used is the Cyberzoo; a flight arena located at the TU Delft, as shown in Figure 4b.

Validation is done for both rotation and translation. For rotation, videos of rotations on the spot are recorded, containing 25 videos in a grid of 5 by 5 meters. The average BoAs, peak ratios and correlation coefficients are computed, as in the simulations presented in the previous section. The results, including the corresponding standard deviations, are shown in Table 2. The first observation is that the BoAs are much smaller than in simulation. This is explained by more spikes (and hence local optima) in the results, which is confirmed by the lower correlation coefficients. It is however, in contrast with the observation in the previous section that more realistic environments yield higher BoAs.

The second observation is that texton and HSV histogram matching show slightly better BoAs than raw pixel matching. Due to the small differences and the large standard deviations however, no significant conclusions can be drawn from this. The corresponding rotation plots are shown in Figure 7.

Table 2: Familiarity performance metrics for each image matching method in the Cyberzoo environment.

	Raw pixels	**Textons**	**HSV**
BoA average	9.13%	12.7%	11.7%
BoA std. dev.	3.38%	6.57%	4.24%
Peak Ratio average	0.53	0.41	0.37
Peak Ratio std. dev.	0.054	0.095	0.093
Corr. Coeff. average	0.82	0.92	0.84
Corr. Coeff. std. dev.	0.093	0.025	0.14

Translation is validated by comparing images taken facing the same direction, in a grid of 49 locations. The results are quite similar to the simulation results and are shown in Figure 8. Again, the result for raw pixel matching shows a very narrow peak at the trained location. This can be disadvantageous for homing, since a small offset from the training path can cause divergence from this path. When looking at

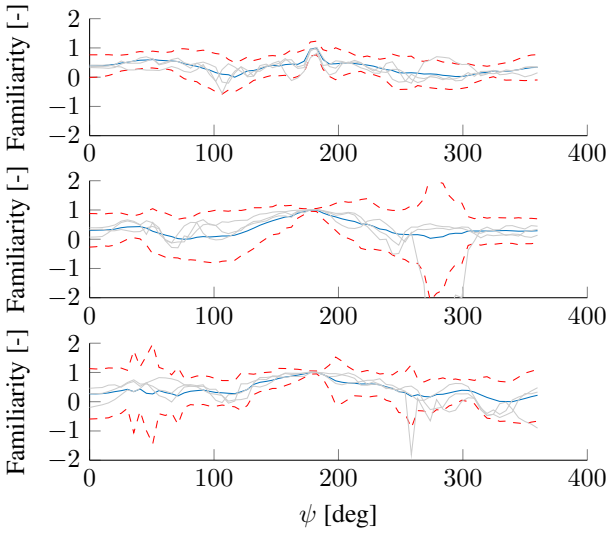


Figure 7: Average rotation on the spot of 25 locations in the Cyberzoo environment. Unfiltered images of 64 by 36 pixels are taken every 5° and compared to a stored image at a heading angle of 180°. The red dashed lines indicate the 2σ bounds and the gray lines are some example familiarities. The three plots indicate raw pixel matching, texton histogram matching and HSV histogram matching respectively.

the texton histogram matching result, it can be seen that two clear optima are present. Even though the surrounding region has quite similar familiarity values, the local optimum at $x = 3$ and $y = 2$ might result in wrong convergence.

Looking at both rotation and translation of HSV histogram matching, it can be observed that the real-life results are better than those made in simulation. This can be explained by more distinct colors in the validation imagery, such that more bins in the HSV histogram are filled.

5 Closed-loop Simulation Flight

As mentioned in the previous sections, the recognition of views during rotation performs best for both raw pixel matching and texton histogram matching. Especially in simulation, the BoAs of these two methods are comparable. When observing familiarity during translations, both texton and HSV histogram matching show a large central region of similar familiarity. As explained earlier, this can be advantageous for homing, since recognizing the correct heading during rotations probably yields the same result for proximate locations. When looking at closed-loop results it is therefore expected that texton histogram matching will perform better than the other two methods.

To show a closed-loop simulation, a simulated robot is placed in the sports hall environment. A route is learned by flying backwards (with a speed of $0.5m/s$), such that the front camera looks in the homing direction, which is necessary to

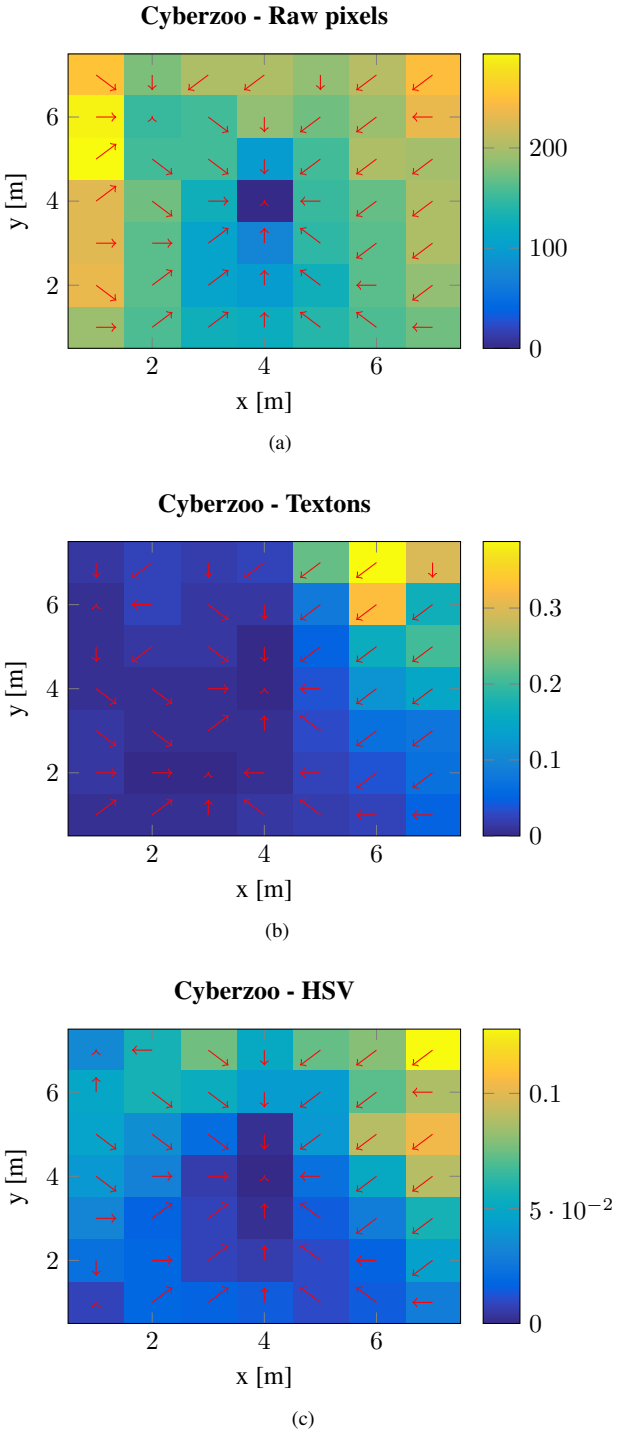


Figure 8: Varying x and y positions using pictures of the Cyberzoo environment, with constant heading angle. Unfiltered images of 64 by 48 pixels are taken in a grid pattern and compared to a stored image at the center of the grid (x=4 and y=4).

use scene familiarity for homing. One third of the image taken at the center is used for training. When homing is initiated, the robot starts flying forward with a constant speed of $0.5m/s$ and the heading is constantly determined using view familiarity. This is done by selecting one third of the image giving the best match with one of the trained views. The center of this image patch is converted to an angle, to which the MAV is steered. Views are obtained from a forward looking camera, with a field of view of $90°$. The result is shown in the left part of Figure 9. Here, the blue solid line is the training route, starting at $x = 4m$ and $y = 12m$, which are arbitrarily chosen. A route of approximatelly $20m$ is flown.

From the results it can be seen that both texton histogram matching and HSV histogram matching approximatelly reach the initial location. The main difference is that texton histogram matching performs turns with a small delay, where HSV histogram matching turns too early. The delay can be explained by low frequency: because all possible patches are extracted from each image, texton histogram matching operates at approximatelly $1Hz$, where HSV histogram matching operates at approximatelly $20Hz$. Texton histogram matching can be significantly improved by using sub-sampling of textons, instead of extracting all. For HSV histogram matching it could be questioned whether it only performs well because the flying direction is approximatelly straight. When homing is done by matching raw pixels (performed at approximatelly $5Hz$), the robot diverges from the trained route. It does, however, follow the curvature of the trained path. The fact that raw pixel matching works worst suggests that differences in familiarity when performing small translational movements causes views to be hard to recognize.

As mentioned, the Infomax neural network can be used as function approximator of familiarity [12]. To test this in closed-loop, the three methods are all represented in a neural network. For both texton and HSV histogram matching a network with 50 inputs is defined (i.e., each histogram forms one input vector to the network). The number of novelty neurons is arbitrarily chosen to be 200. Furthermore, the number of epochs is set to 500. It turned out that a lower number of epochs gives significantly worse performance. In further simulations or flight tests this should be tuned by testing multiple numbers of both novelty neurons and epochs. For raw pixel matching, the image is scaled down to 16 by 12 pixels, which gives 192 inputs to the network. Larger dimensions as input cannot be performed in real-time. The number of novelty neurons and epochs are kept the same.

The results using an Infomax network can be seen in the right part of Figure 9. It is clear that the results are slightly worse than with a perfect memory (i.e., by keeping a database of images, texton histograms or HSV histograms). It does however, look quite similar to the perfect memory case, which suggests that the assumption that Infomax is only used as approximator for views is quite good.

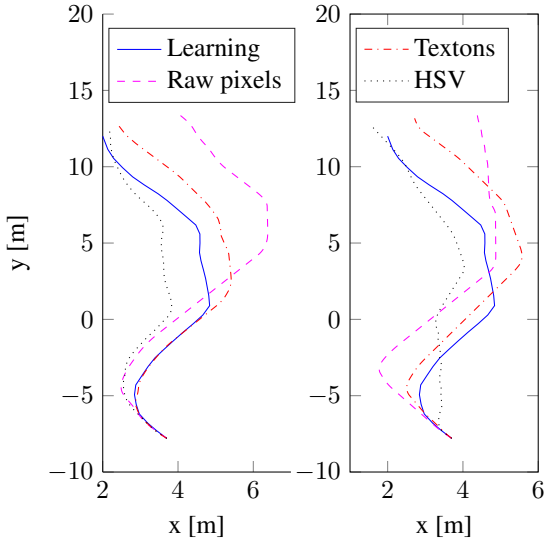


Figure 9: Closed-loop homing simulation in the sports hall environment in SmartUAV. On the left, a perfect memory is used; on the right the Infomax neural network is applied.

6 Discussion

When first looking at the rotational analysis, it was observed that raw pixel and texton histogram matching performed best. When looking at the translation results, raw pixel matching shows the most distinct peak. Because position of the robot is not directly controlled, it is advantageous that a large familiar region appears in translation, so that a small displacement of the robot does not change the homing performance. This was especially the case for texton histogram matching and HSV histogram matching. This suggests that texton histogram matching would perform best, which is confirmed by the closed-loop results. Surprisingly, HSV histogram matching shows very good performance in closed-loop. A reason for this can be that generating and storing HSV histograms is computationally very efficient, which allows for a low timestep. This means corrections are made very quickly so that the robot does not diverge too much. It does not say however, that HSV histogram matching would perform well when divergence already happened.

When evaluating the closed-loop tests in this paper, some limitations can be identified. First of all, it is only tested in simulation. Although the fidelity of the simulation is higher than the simulations performed by Baddeley et al., it is questionable whether the same results would be obtained in a real flight. Furthermore, additions can be proposed to make the algorithm more robust. An example is to use active rotation instead of using the inherent field of view of the forward looking camera, such that bigger turns can be made. Alternatively, a camera with a larger field of view can be added. Another possibility is the use of visual odometry to get a rough esti-

mate of the path taken. Odometry could be used to prevent severe divergence from the correct route. Since the experiment enforces small turns only, it cannot yet be concluded that the method works well for diverse trajectories.

Another point of discussion is that the main reason scene familiarity can be a viable approach for visual homing of MAVs is computational efficiency. The only way this is tested in this paper, is by performing closed-loop real-time simulations on a laptop computer. When implementing the algorithm on-board an MAV, the real-time performance may be inadequate due to a slower micro-processor. The one exception was HSV histogram matching, because both the computations needed to extract histograms, and the storage capacity are limited. In this paper however, all textons were extracted from each image. Usually, it suffices to randomly pick a set of textons, which would drastically improve computational performance. The storage of a texton histogram is similar to storing an HSV histogram. A huge advantage of using a neural network is that the storage capacity is constrained. Even though this means that the network can *forget* earlier trained views (which is also investigated by Baddeley et al.), it allows control over the often very limited storage capacity on MAVs. Training on the other hand, is quite slow; especially when having to train each sample 500 times.

7 Conclusion and Recommendations

This paper investigates the applicability of the scene familiarity homing method, observed from insect behavior, to MAVs. The scene familiarity method is introduced as proof of concept for desert ants to use the recognition along a route to find their way home. Next to this, an unsupervised neural network was used to keep storage of familiarity compact.

The concept of only using recognition along a route is a very interesting one. The analysis shows the closed-loop performance is good. The reason the method is promising, is the computational efficiency. Especially HSV histogram matching showed surprisingly good closed-loop performance while running quite fast. For the other two image representations the algorithm works in real-time on a laptop, although the frequencies in the current implementations are low.

It is concluded that using texton or HSV histogram matching is useful for visual homing on small robots. Once a route is lost, the risk of divergence is quite high. This must be further investigated. It seems very useful to combine scene recognition with existing methods like visual odometry. This way, two computationally efficient algorithms can be combined to succesfully perform homing.

References

[1] Z. Mathews, M. Lechon, J.M. Blanco Calvo, A. Dhir, A. Duff, S Bermudez i Badia, and P.F.M.J. Verschure. Insect-like mapless navigation based on head direction cells and contextual learning using chemo-visual sensors. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, New York, USA, oct 2009. IEEE.

[2] M.V. Srinivasan. Where paths meet and cross: navigation by path integration in the desert ant and the honeybee. *J Comp Physiol A*, 201(6):533–546, may 2015.

[3] R. C. Nelson. Visual homing using an associative memory. *Biological Cybernetics*, 65(4):281–291, aug 1991.

[4] B. Baddeley, P. Graham, P. Husbands, and A. Philippides. A model of ant route navigation driven by scene familiarity. *PLoS Computational Biology*, 8(1), jan 2012.

[5] B.D.W. Remes, P. Esden-Tempski, F. Van Tienen, E. Smeur, C. De Wagter, and G.C.H.E. De Croon. Lisa-s 2.8g autopilot for gps-based flight of mavs. Delft University of Technology, 2014.

[6] E. Motard, B. Raducanu, V. Cadenat, and J. Vitria. Incremental on-line topological map learning for a visual homing application. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, New York, USA, apr 2007. IEEE.

[7] B. A. Cartwright and T. S. Collett. Landmark learning in bees. *Journal of Comparative Physiology*, 151(4):521–543, 1983.

[8] T. S. Collett. Insect navigation en route to the goal: Multiple strategies for the use of landmarks. *Journal of Experimental Biology*, 199:227–235, 1996.

[9] D. Lambrinos, R. Möller, T. Labhart, R. Pfeifer, and R. Wehner. A mobile robot employing insect strategies for navigation. *Robotics and Autonomous Systems*, 30(1-2):39–64, jan 2000.

[10] D. Lambrinos, R. Möller, R. Pfeifer, and R. Wehner. Landmark navigation without snapshots: the average landmark vector model. In *Proceedings of Neurobiology Conference Göttingen*, 1998.

[11] J. Zeil, M.I. Hofmann, and J.S. Chahl. Catchment areas of panoramic snapshots in outdoor scenes. *Journal of the Optical Society of America A*, 20(3):450, mar 2003.

[12] A.J. Bell and T.J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, nov 1995.

[13] D.D. Gaffin and B.P. Brayfield. Autonomous visual navigation of an indoor environment using a parsimonious, insect inspired familiarity algorithm. *PLOS ONE*, 11(4):e0153706, apr 2016.

[14] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *Int J Comput Vision*, 62(1-2):61–81, apr 2005.

Chapter 4

Literature Overview

UAV navigation is an ongoing research topic. Many different approaches are being used to find solutions to specific or more general navigational problems. This chapter reviews earlier work done in finding a solution to the navigation problem, and specifically a solution to the homing problem, as stated in the introduction. The focus will be on vision-aided navigation, mainly because a camera sensor is cheap, light-weight, versatile, and it is already available on the test platforms.

In this thesis, two definitions for navigation are distinguished (Franz & Mallot, 2000):

i) An answer to the three questions: Where am I? Where are other places relative to me? How do I get to other places from here? (Levitt & Lawton, 1990)

ii) Navigation is the process of determining and maintaining a course or trajectory to a goal location. (Franz & Mallot, 2000; Gallistel, 1990)

The main difference between those definitions is that the second definition does not set the requirement for the agent to know some sort of map, including its own location. Instead, it specifies that an agent must be able to reach a behavioral goal, regardless of the layout of the environment. Navigation where self-localization of an agent with respect to the goal location is used, is in the context of insect navigation referred to as true navigation (Graham, 2010).

Especially the second definition is useful in the context of this thesis, as the efficiency of an algorithm is considered to be more important than having real-time information on the current location.

Homing is a very limited part of the navigation problem. The only goal is to return back to the initial location as efficient as possible. Knowledge about the location of the vehicle and its environment can be useful, but will always be a means to reaching the home location. Therefore, the second definition for navigation will be used throughout this thesis.

The chapter is split up into two parts. First, in Section 4-1 an overview is given of visual navigation techniques applied in robotics and specifically on UAVs. Then, research that has investigated insect navigation and homing methods is presented in Section 4-2, which includes implementations in robotics as well.

4-1 Vision-Based UAV Navigation

Using computer vision to navigate a UAV is an active field of research. The availability of cheap cameras with a small footprint and the growing computational power of Micro Processing Units (MPUs) make real-time camera usage for navigation possible.

Most vision-aided navigation research is done on the navigation problem in general. This section gives an overview of different navigation solutions. As homing is a small part of this navigation problem, these techniques can also be applied to this.

The different methods are divided in three different categories, as shown in Figure 4-1 (Wolf, 2011). These categories will be used as guideline to go through the different methods.

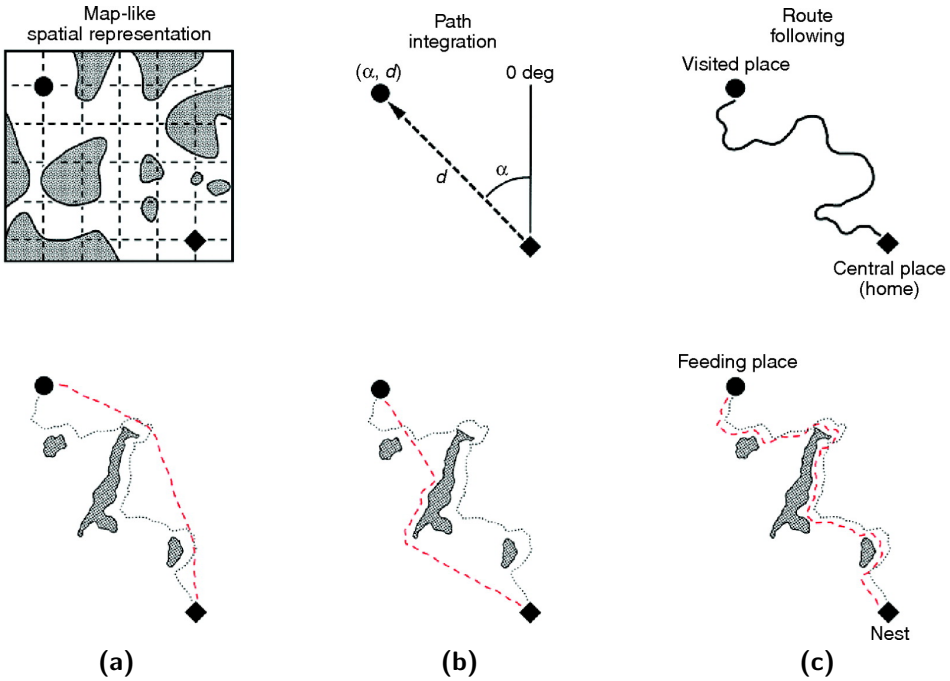


Figure 4-1: Three different modes of navigation: map-like representations (4-1a), path integration (4-1b) and route following (4-1c). The top pictures illustrate the navigation principles and the bottom pictures contain a homing example using that method. Adapted from (Wolf, 2011).

First, section 4-1-1 shows the use of methods to perform localization in a generated map. After this, section 4-1-2 focusses on Visual Odometry (VO), where optic flow is integrated to obtain the traveled distance.

Then, the focus will switch to insect-inspired methods, which will continue with Path Integration (PI) methods and also introduce route familiarity.

4-1-1 SLAM

Map-building methods perform concurrent map-building and localization within that map. This is referred to as Simultaneous Localization and Mapping (SLAM) (Leonard & Durrant-Whyte, 1991). SLAM is a paradigm in which the agent generates a map when traversing a

route, and simultaneously localizes itself and navigates on this map. This means that SLAM in principal consists of three parts, namely traversing/exploring a route, constructing a map and self-localization and navigation on this map (Bonin-Font, Ortiz, & Oliver, 2008).

Currently, both the theory behind SLAM and the implementation of different methods are researched actively. In (Dissanayake, Newman, Clark, Durrant-Whyte, & Csorba, 2001) the authors show that a general SLAM solution must exist, by proving that the determinant of the covariance matrix of the map estimation problem decreases with successive observations. The availability of a theoretical proof of concept of the SLAM problem is confirmed in (Durrant-Whyte & Bailey, 2006), with the notion that a practical realization of a general SLAM solution still contains many issues. These issues mainly concern environmental conditions, like lighting and temporary occlusions.

SLAM solutions must deal with three key problems, which Fuentes-Pacheco et al. refer to as data association problems. These problems are (Fuentes-Pacheco, Ruiz-Ascencio, & Rendón-Mancha, 2012):

- **Loop closure detection** can correct errors in a generated map by re-visiting a mapped location (Ho & Newman, 2007; Clemente, Davison, Reid, Neira, & Tards, 2007; B. P. Williams, 2009). A problem originating from this is perceptual aliasing, where similar views of different locations cannot be distinguished (Angeli, Doncieux, Meyer, & Filliat, 2008b).
- **Robot kidnapping** addresses the situation where a robot performing SLAM is moved to another location within the map, without knowing where it is placed. This can also occur when the robot moves blindly due to occlusions or sensor outage (Eade & Drummond, 2008; B. Williams, Klein, & Reid, 2011).
- **Multi-session mapping** refers to the situation where a map made of a part of the environment during an earlier SLAM session must be reused. This earlier generated map will be aligned with maps made during later sessions in different parts of the environment (Ho & Newman, 2007). This problem is similar to the case where multiple robots perform SLAM simultaneously in the same environment (Gil, Reinoso, Ballesta, & Juliá, 2010; Vidal-Calleja, Berger, Solà, & Lacroix, 2011).

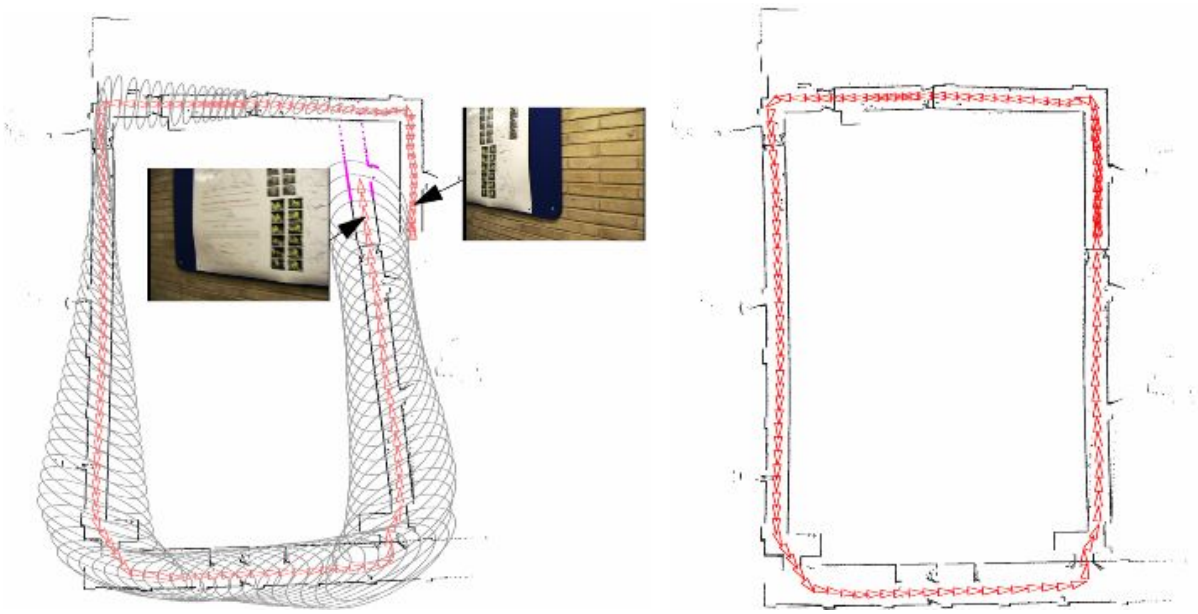
Especially when localization and mapping are done using a camera, performing it on-board is computationally demanding. As both map construction and localization need to be done in real-time, the application of on-board visual SLAM on small UAVs is quite new.

A vast number of SLAM algorithms are presented in literature. An overview of methods can be found in (Fuentes-Pacheco et al., 2012, Table 1).

The maps made in SLAM solutions can be divided into two categories: grid-based (metric) SLAM, where dense photographic maps are made, and topological SLAM, which only maps certain landmarks and connections between them (Thrun, 1998). First metric SLAM based on probabilistic filters is described, after which several topological methods are presented.

EKF SLAM

Most SLAM solutions are based on probabilistic filters. In (Smith, Self, & Cheeseman, 1990) a map-building approach is presented using an Extended Kalman Filter (EKF). An EKF is a



(a) **(b)**

Figure 4-2: Example of a SLAM loop closure, where 4-2a shows the map before loop closure and 4-2b shows the closed map. Extracted from (Ho & Newman, 2006).

recursive filter, which integrates a state transition model and an observation model in order to get a state (and in this case also map) estimate.

A notable EKF SLAM homing solution is shown by the "explore and return" experiment published in (Newman, Leonard, Tardos, & Neira, 2002), where a ground robot performs mapping during manually controlled exploration and autonomously returns to the initial location. During manual exploration, the operator only used real-time generated maps. Processing of the algorithm was done on a laptop computer.

The presented algorithm can be implemented for different robots and sensors. In (Newman et al., 2002) odometry and laser ranging is used. Using a camera instead will increase the computational demands of the algorithm. This and the fact that the algorithm is run off-board, violate the research criteria specified in the introduction, making the solution too demanding for the homing problem posed in this thesis.

Another EKF SLAM solution is presented in (Magree & Johnson, 2015), where a numerically stable visual SLAM algorithm is introduced and applied to a Yamaha R-Max unmanned helicopter[1] (the same algorithm is implemented on a light-weight quadrotor in (Magree, van Dalen, Haviland, & Johnson, 2015)). The algorithm is, in combination with an Inertial Measurement Unit (IMU), used for both stabilizing the vehicle and performing localization. Unlike full-blown SLAM solutions, only a local map (i.e., close to the current helicopter location) is kept up-to-date. This makes real-time performance quite good, although the presented results are performed on a high-end Core i7 processor. As no overall map is maintained, the data association problems are not accounted for, which gives rise to propagating errors in localization.

[1] `http://rmax.yamaha-motor.com.au/`

Disadvantages of EKF based methods are the possibility of divergence when a wrong measurement-covarience combination is provided and the high computational complexity with increasing map size.

Topological SLAM

The difference between metric SLAM and topological SLAM is found in the representation of the environment. A purely topological map is an abstract representation of the environment containing nodes and connections between them. This is illustrated in Figure 4-3, where an office environment is represented by a series of connected nodes.

Using this representation, only the most significant places are mapped and connected to each other (Fraundorfer, Engels, & Nister, 2007; Eade & Drummond, 2008). This makes it computationally more advantageous than metric SLAM, as only small parts of the environment have to be mapped in detail. However, to make sure the map is robust, global optimization is needed in order to keep localization errors small (Frese, Larsson, & Duckett, 2005; Olson, Leonard, & Teller, 2006).

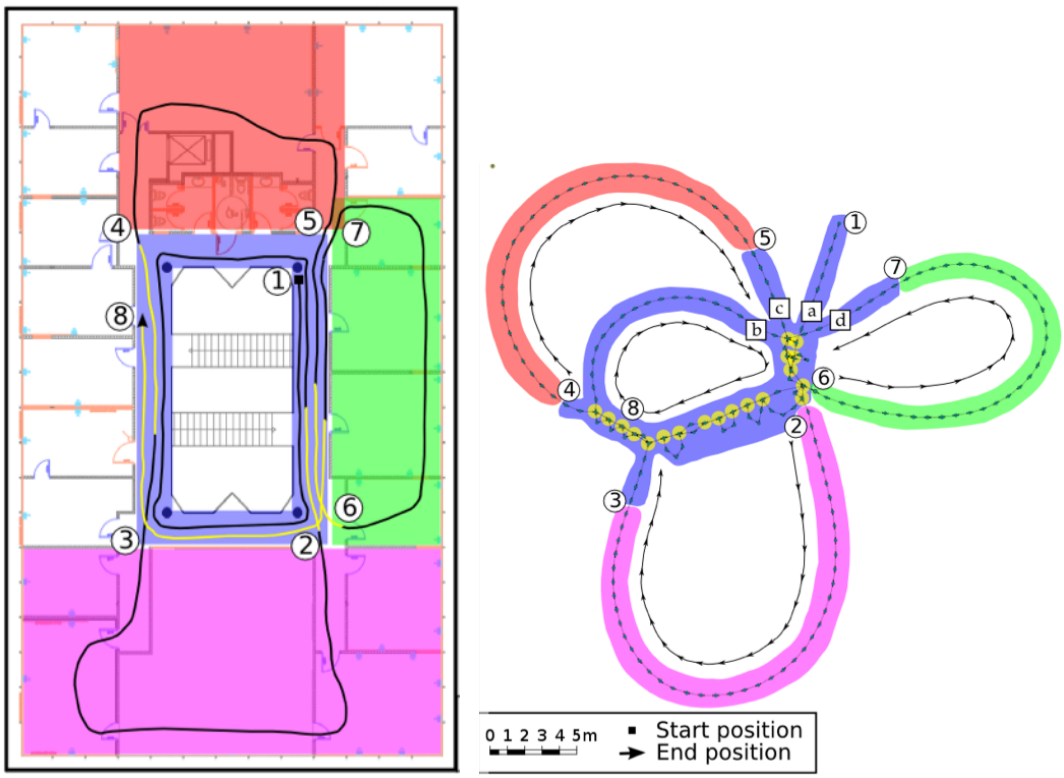


Figure 4-3: Example of a topological map, where 4-3a shows an office environment and 4-3b shows the corresponding map. Extracted from (Angeli et al., 2008a).

The idea behind using a topological mapping for SLAM methods is introduced in (Kuipers & Byun, 1991), where it is described as a graph of distinctive places and travel edges. An example definition of such graphs are Generalized Voronoi Graphs (GVGs) introduced in (Choset,

2000).

In (Garcia-Fidalgo & Ortiz, 2015) topological maps are classified in three categories:

- **Global descriptors** use full images to describe nodes in maps and are therefore fast. An overview of SLAM methods using those descriptors can be found in (Garcia-Fidalgo & Ortiz, 2015, Table 2).
- **Local Features** can be used to only describe distinctive parts of the image, instead of the entire image. To make sure local features form a robust representation of a view, (Tuytelaars & Mikolajczyk, 2007) described the following properties of good local features: repeatability, distinctiveness, locality, quantity, accuracy and efficiency.
- **Visual Bags of Words (BoWs)** contain vocabularies of image patches. An image can be represented by a histogram of occurrences of those patches (Garcia-Fidalgo & Ortiz, 2015, section 4). The BoW method originates from counting the number of words in documents using a predefined vocabulary. In (Eade & Drummond, 2008) BoWs are used to identify loop closures in a real-time monocular topological SLAM algorithm.

Current topological SLAM implementations often use less abstract representations, where the nodes contain geometric information. A mapping strategy based on this approach is published in (Tapus & Siegwart, 2005) and uses fingerprints to represent nodes (Lamon, Nourbakhsh, Jensen, & Siegwart, 2001). In the context of topological SLAM fingerprints can be any representation of a mapped node. Lamon et al. represent each fingerprint as a string which encodes color patches and vertical edges to describe the location. A map is pre-built and consists of a fixed amount of fingerprints. Localization is done by a minimum energy optimization between mapped fingerprints and fingerprints obtained at the current location of a robot. In (Tapus & Siegwart, 2005) this method is extended by an incremental mapping approach based on these fingerprints. A combined SLAM system is found in (Tapus, 2005).

In (Motard, Raducanu, Cadenat, & Vitria, 2007) an incremental topological map learning algorithm is proposed, specifically aiming towards visual homing. The algorithm is designed for the AIBO robot[2] and is meant for the robot to navigate back to a charging station when needed. AIBO is already capable of navigating back towards the charger when it is within a range of one meter from it. The goal of the proposed topological SLAM algorithm is to navigate the robot to within a meter from the charger. The approach makes use of two modules. The first module is used for online incremental map-learning during exploration. The second module is for route planning and is only activated when the robot needs charging.

For map learning, Motard et al. make use of a graph consisting of Scale Invariant Feature Transform (SIFT) features (Lowe, 2004) and their spatial relationships. These locations are connected so that a chain of snapshots represents the environment. When a new topological location is recorded which is sufficiently distinct from previous ones, it is added to the chain of snapshots. Besides distinctiveness, another requirement for new snapshots is that it has to be connected to an earlier one, as a camera is the only sensor used. A homing solution is calculated using a shortest path algorithm (Dijkstra's algorithm (Cormen, Leiserson, Rivest, Stein, et al., 2001)).

[2] `http://www.sony-aibo.co.uk/`

The AIBO experiment is performed in real-time, but image processing is done on an external PC (i.e., frames are sent to the PC for on-the-fly processing) at a frequency of 10 frames/s. This suggests that the algorithm is computationally expensive. Experimental results are good for the object rich test location. The robustness of the algorithm in larger environments is not yet guaranteed.

Hybrid SLAM

Hybrid SLAM combines the best of metric and topological SLAM. Usually it consists of connected nodes, where the nodes are represented by a metric map. This combines the accuracy of metric SLAM and the efficiency of topological SLAM (Thrun & Bü, 1996).

An example of hybrid SLAM is RatSLAM (Milford, Wyeth, & Prasser, 2004). RatSLAM is a bio-inspired method that models navigation abilities found in the hippocampus of rodents. Especially the observation that rats have place fields in their brains, which are patterns of neural activity, induced by moving and visual inputs, defines the difference between RatSLAM and topological SLAM solutions. Which fields are active determines the rat's knowledge of its current location. In RatSLAM these place fields are implemented as pose cells, which are competitive attractor Neural Networks (NNs). These networks contain local clusters of activation, so that certain inputs activate a clustered part of the network. The total activation of a network is constant so that the pose cells form a probability distribution of location or heading. These pose cells act as topological landmarks with metric SLAM behavior. The location of cells in the competitive attractor NN gives information about the location of the robot, which can be considered metric. In RatSLAM two NNs are implemented: one for heading and one for the two dimensional location of the robot. As sensory input the algorithm combines wheel odometry with vision, which detects coloured cylinders and outputs the distance and bearing from the robot to that cylinder.

The fact that RatSLAM is bio-inspired, makes it especially interesting for this thesis. Still, the system was tested on an external computer and is quite demanding. As with metric SLAM solutions, the computional price paid for the more diverse navigational abilities does not comply with the requirements specified in the introduction.

Conclusion

As mentioned in the introduction the purpose of this thesis is to find a homing algorithm for an MAV. The SLAM methods presented in this section can do much more than what is needed for visual homing. If a map is generated, navigation between different points in the map can be done. The cost of this is a high computational demand of the algorithms.

4-1-2 Visual Odometry

Path Integration (PI) methods form navigation solutions where localization at different moments in time are used to navigate. A velocity is obtained and integrated, to perform localization. In principle, no map is made of the environment, although when heading estimation is part of the localization, the result will be a map-like representation. The major difference with SLAM is that the data association problems are not accounted for.

In robotics, velocity is often obtained using optic flow between different frames of a scene (Desouza & Kak, 2002). Optic flow is the apparent motion observed in an image, which Chao et al. define as a two dimensional projection of three dimensional relative motion in an image (Chao, Gu, & Napolitano, 2013). Many different algorithms exist to calculate optic flow. A comparison of these methods and different UAV applications can be found in (Barron, Fleet, & Beauchemin, 1994) and (Chao et al., 2013) respectively. For UAV applications, the most common algorithms for calculating optic flow are the Lucas-Kanade (LK) method (Lucas & Kanade, 1981), the Horn-Schunck method (Horn & Schunck, 1981), image interpolation methods (Srinivasan, 1994), block matching techniques and feature matching techniques.

Optic flow is not only used for UAV navigation, but also for lower level operations like obstacle avoidance and autonomous landings. For the homing problem posed in this thesis, the aim is to record and store the traversed route of an MAV. This can be done with optic flow, by estimating the velocity vector of the vehicle and integrating this to obtain position. In (Ding et al., 2009), a height and velocity estimation technique using optic flow is used when GPS fails temporarily. Optic flow is used in combination with an Inertial Navigation System (INS) and is integrated in an EKF. It is shown that the measurements are noisy, but are useful for short periods of time.

As mentioned, this localization is referred to as Visual Odometry (VO), which formally entails the motion estimation of a mobile robot using only camera images as sensory input. The term was first introduced by (Nister, Naroditsky, & Bergen, 2004) and is based on odometry of ground vehicles, where wheel rotations are used to measure motion. VO is a form of Structure from Motion (SfM), but only focuses on motion estimation instead of map construction. In this thesis the term VO is used more loosely: besides using cameras, other sensors can aid the path estimate.

Research in VO is done for both stereo vision and monocular vision. When stereo vision is used, it is easier to make a three dimensional representation of the environment due to available depth information. For monocular vision, this depth information must be extracted from sequential images. This means that stereo VO can operate without motion, where in the monocular case motion is needed to extract environmental geometry.

In (Nister et al., 2004) both stereo and monocular VO approaches are presented. In both cases Harris corners (Harris & Stephens, 1988) were identified in each frame, and matched by computing correlation of image patches around each corner. From the matched frames, the geometric motion was extracted using the 3-point algorithm (Haralick, Lee, Ottenberg, & Nölle, 1994) and preemptive Random Sample Consensus (RANSAC) (Nistér, 2005).

In (Nister et al., 2004) an experiment with a ground robot is presented, where a comparison is made between VO and differential GPS. The results are very good, since a small accumulation of drift is observed during these tests (i.e., a position error of one to five percent over less than 300 meters). A comparable but more recent research is published in (Strydom, Thurrowgood, & Srinivasan, 2014), where a quadrotor navigation system is designed using a combination of optic flow and stereo vision.

Currently, many VO algorithms (for example, (Blosch, Weiss, Scaramuzza, & Siegwart, 2010; Weiss et al., 2013)) are based on Parallel Tracking and Mapping (PTaM). PTaM (Klein & Murray, 2007) is a SLAM system that separates feature tracking from creating a map of 3D

features. The tracking uses a large set of low quality (but computationally efficient) features, to construct a dense map of a small environment. This does not make it suitable as full blown SLAM navigation system, but it can be used as an accurate form of local VO.

Semi-Direct Visual Odometry

Most VO method make use of feature tracking to get a velocity and position estimate. Extraction of features is computationally expensive. A recently published approach is Semi-direct Visual Odometry (SVO), which only extracts features on so-called keyframes (Forster, Pizzoli, & Scaramuzza, 2014). These are frames which contain significant changes in the scene. Between two keyframes, matching is done using pixel intensities which is significantly faster than extraction and matching of features. The combinations of feature matching and pixel (gradient) matching make SVO a fast and accurate three dimensional pose estimator.

Forster et al. did multiple experiments, among which a flight test with an MAV with a downward looking camera. Processing was done on-board, on an Odroid-U2 (ARM Cortex A9, 1.6 GHz, 4 cores), and two cores were used for the algorithm (one for motion estimation and one for updating the keyframe database). The algorithm ran with a speed of approximately 50 Hz, and had a position Root-Mean-Square (RMS) error of 0.0059 m/s, and a rotation RMS error of 0.43 deg/s. The speed and accuracy increase of SVO over other VO algorithms is mainly due to the use of a depth filter, where outlying features are rejected.

Conclusion

From the different methods it can be concluded that VO is quite accurate. The inherent integration errrors are small; especially for indoor homing, where distances are small. The accuracy is related to the computional complexity of algorithms. For instance SVO is very accurate but computationally quite demanding for more accurate settings. When ample computational resources are available, the homing problem can be solved with VO. The next section will show insect-inspired homing methods, where VO often plays a part, but with even less computational resources. This means different strategies are needed to obtain the same accuracy.

4-2 Insect-Inspired Homing

The problems with SLAM solutions as described in the previous section are the computational and memory requirements imposed by such algorithms. Especially for metric SLAM, maps can get very big when the distance covered by the robot increases. The suitability of such algorithms is limited for the use of MAVs, as the computational demands are quite high. Mapless methods, however, are much more efficient, but induce velocity integration errors over time. Even though these errors are not always significant, a trade-off has to be made between accuracy and speed.

Looking into biology, many insects face the same navigational problems as UAVs. Especially visual homing is considered to be an important skill for insects like honeybees and ants (T. S. Collett, 1996; T. S. Collett & Collett, 2002). Since the toolbox available to insects

in order to navigate successfully is limited, much research is done in navigation habits of ants and bees, from which computationally efficient algorithms can be created. This section gives an overview of insect navigation research and also presents robotic implementations.

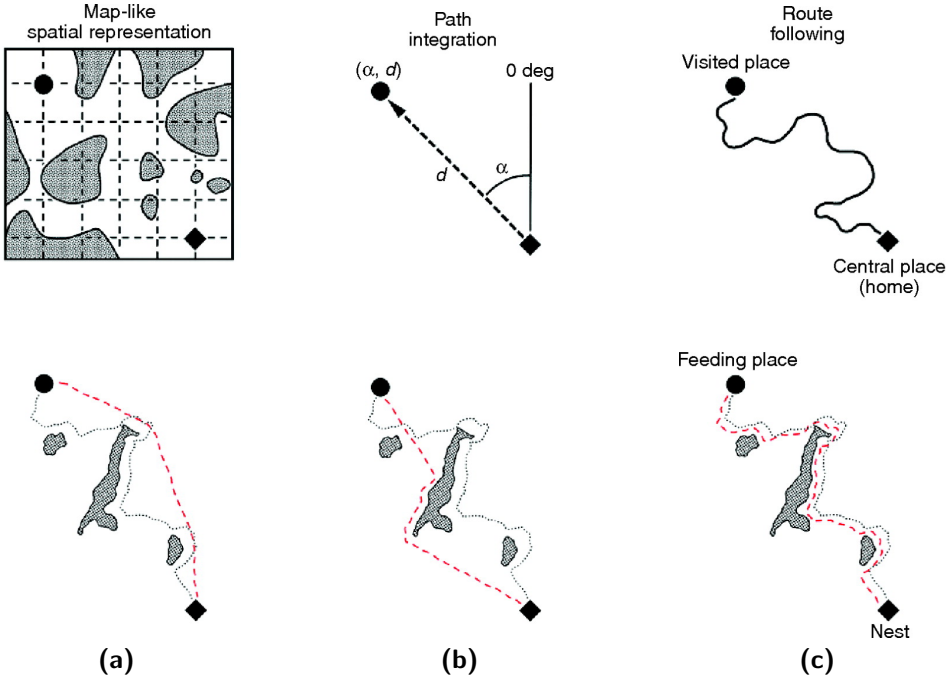


Figure 4-4: Three different modes of navigation: map-like representations (4-4a), path integration (4-4b) and route following (4-4c). The top pictures illustrate the navigation principles and the bottom pictures contain a homing example using that method. Adapted from (Wolf, 2011).

As mentioned, the structure in this chapter is based on the navigation modes repeated in Figure 4-4 (Wolf, 2011). The previous section already covered map-based representations and path integration. The latter is continued in this section.

section 4-2-1 discusses research on optic flow observed in honeybees and MAV navigation techniques which make use of this. Then, section 4-2-2 goes into the famous snapshot model as homing method, and section 4-2-3 describes the use of landmark vectors to find a trajectory to a home location.

After this, Section 4-2-4 presents several views on the question whether insects make use of maps for navigation or not. Finally, Section 4-2-5 shows a modern route learning method (Figure 4-4c) as alternative to landmark methods like the snapshot model.

4-2-1 Visual Motion Detection

In the previous section mapless navigation techniques are presented, which make use of optic flow. Several researchers observed visual navigation based on optic flow in honeybees. Therefore, this section gives an overview of visual motion detection employed by honeybees.

Research has shown that bees make use of visual cues to navigate between food sources. In (Srinivasan, Zhang, Lehrer, & Collett, 1996) an experiment is presented with the aim to

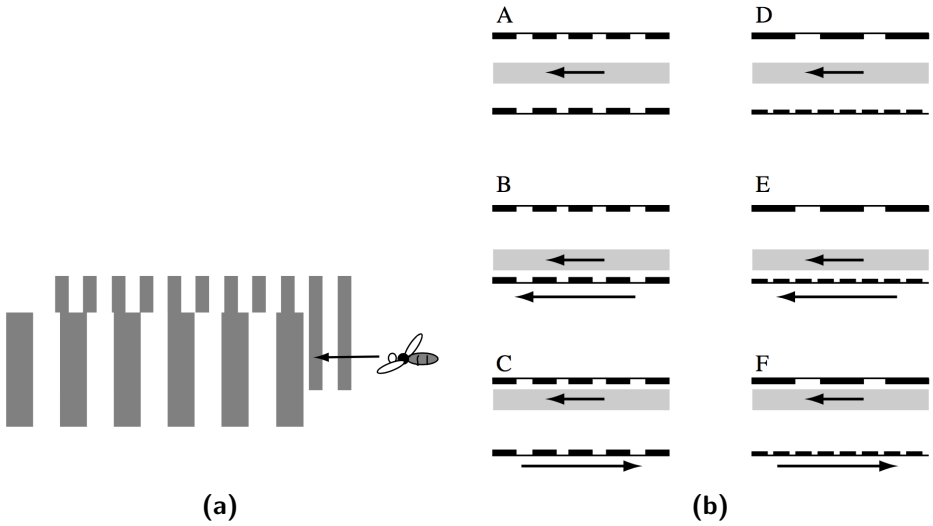


Figure 4-5: Experiment shows that bees obtain range from apparent image speed. The short arrows indicate direction of flight and the long arrows the movement of the images on the wall. Rearranged image copied from (Srinivasan et al., 1996).

test the observations that bees always fly through the center of a confined space and that they know when to stop moving. The hypothesis is that the motion in retinal images is used to estimate velocity on all sides. To test this hypothesis an experiment is set up where honeybees fly through a tunnel towards a food source. The walls of the tunnel have a black-and-white striped pattern (Figure 4-5).

To find out whether apparent motion is indeed used by honeybees to fly through the center of the tunnel, the patterns in the tunnel are moved asymmetrically. As illustrated in Figure 4-5b honeybees change their location in the tunnel such that the apparent motion is constant on both sides. Different dimensions of the black-and-white blocks on both sides of the tunnel did not influence the flying behavior of the honeybees.

The second hypothesis is that honeybees also control their flight speed with apparent retinal motion. This was tested in (Srinivasan et al., 1996) by letting bees fly through a tapered tunnel (i.e., changing width of the tunnel). It turned out that the bees accelerated when the tunnel got narrower and decelerated when the tunnel got wider. This suggests the use of apparent motion to control the flight speed.

Further experiments confirm that honeybees apply VO using optic flow to estimate when to stop moving and hence to navigate (Esch & Burns, 1996; Srinivasan, Zhang, & Bidwell, 1997). A more detailed overview of experiments done to study the VO capabilities of honeybees can be found in (Srinivasan, 2014). The use of VO by integrating optic flow for motion estimation is already used frequently in UAV navigation, as mentioned in earlier sections of this chapter.

A biologically-inspired application of the use of optic flow for robot homing is presented in (Diamantas, Oikonomidis, & Crowder, 2010). As seen with topological SLAM methods, fingerprints are used as landmarks and are represented by LK optic flow vectors. A training algorithm is presented where the optic flow fingerprints are stored when traversing a route. The platform contains two cameras on the sides of the vehicle, which record optic flow vectors

that are a function of the distance from the camera to the object and the speed. A fingerprint can thus be seen as a sequence of optic flow vectors. During homing the currently obtained sequence of optic flow vectors is compared to the fingerprints in the topological map. Based on a similarity threshold a probability that it is indeed the correct landmark is produced. During simulations the authors show that a similarity of 20% between optic flow vectors perceived during homing and optic flow in a given fingerprint is enough to correctly identify a landmark.

4-2-2 Snapshot Model

In 1983, Cartwright & Collett introduced the Snapshot Model (Cartwright & Collett, 1983). The framework they presented gives an explanation of the navigation capabilities of bees when traveling between different food sources. The model consists of two main elements: a dead-reckoning method to get close to the goal location and finding the best visual match with a stored snapshot in order to find the exact location of this goal. The visual matching is done by a direct comparison of an image on the retina with a stored snapshot.

For highly cluttered environments close to the goal location, the visual matching would only work when the distance is very small. In (Cartwright & Collett, 1987) the snapshot model is extended by adding an extra snapshot, which does not contain landmarks close to the goal. This snapshot can be used for visual matching when the distance to the goal is larger, while the other snapshot (including visual information close to the goal) can be used for the last part of the homing route. Navigation based on those snapshots is done by comparing size and azimuth of the landmarks between the snapshot and retinal image (Franz, Schölkopf, Mallot, & Bülthoff, 1998).

In (T. S. Collett, 1996) the landmark approach is further extended by the addition of visual beacons. During the early parts of the return to a feeder or nest location, not only dead-reckoning but also visual landmark information is used, so that they serve as beacons which can correct heading errors.

In order to make the snapshot model biologically more plausible, (Möller, Maris, & Lambrinos, 1999) present a neural implementation of the snapshot model. It gives an explanation of why and how the snapshot model can work in an insect's brain. They show that even though the real neural implementation in an insect's brain is unknown, a simple neural model can mimic snapshot homing.

Since the introduction of the snapshot model in a biological context, many successful attempts have been done to use this model in robot navigation. In (Argyros, Bekris, Orphanoudakis, & Kavraki, 2005) a method is proposed where navigation happens between Milestone Positions (MPs). A navigation problem is split up into multiple local navigation tasks from one MP to the next. An MP is specified by a set of at least three image features (Shi & Tomasi, 1994) and a new MP is defined when these features (partly) disappear from view. The images are taken with an omni-directional panoramic camera. An illustration of homing in a grid using multiple MPs can be seen in Figure 4-6.

Navigation between two MPs happens by calculating a motion vector based on three or more image features. This vector is an average of partial vectors drawn for each possible feature pair, and is updated constantly. The construction of a partial motion vector is illustrated in

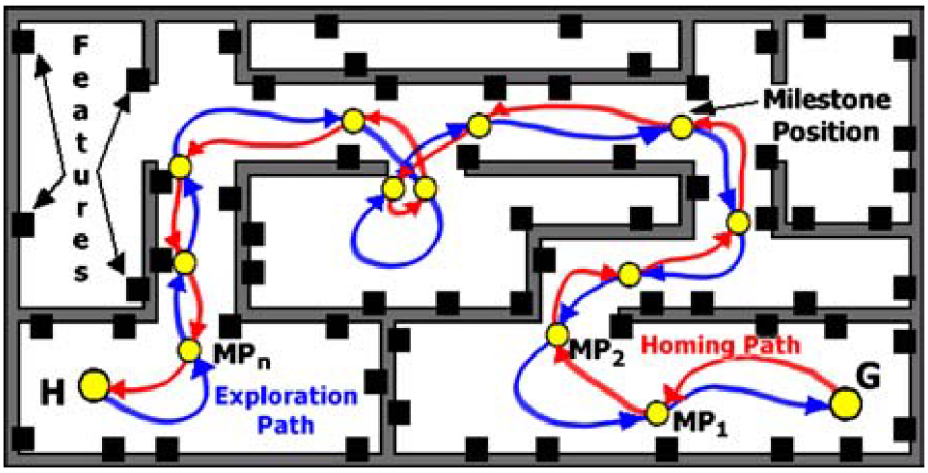


Figure 4-6: Long-range homing strategy using multiple MPs. H indicates the home location and G is the goal location from which homing is initiated. Obtained from (Argyros et al., 2005).

Figure 4-7. In this angular approach two mirrored hyperbolas are drawn with two features as focal points. The tangent at S of the hyperbola going through the location of the agents forms the partial motion vector. The angular approach is tested on a slow-moving ground robot in an office environment, where a homing accuracy in the order of centimeters was achieved.

Another implementation of the snapshot model for robot homing is presented in (Pons, Hübner, Dahmen, & Mallot, 2007), which is derived from a vision-based homing method published in (Vardy & Moller, 2005). The goal is to improve their homing algorithm, by taking dynamic changes of the environment into account. This means occlusions and illumination changes are considered in the experiment. Robust landmarks are defined as a representation of SIFT features, and a matching and voting scheme is used to determine which features describe the location best (Vardy & Moller, 2005).

The experiments were performed on an all terrain four-wheel drive ground robot, with an onboard laptop with dual core processor for computations and an omnidirectional camera. Both an indoor and an outdoor experiment were performed, with varying brightness. Performance was measured by evaluating the estimated homing vectors and the success rate of homing after a fixed amount of movements. In the indoor experiment it was observed that increasing occlusions and illumination did not affect the mean homing vector, but increased the standard deviation. When the home location was placed far away from the place where homing was initiated, the performance decreased drastically under the influence of dynamic changes in the environment. The outdoor experiments were performed at different times during the day to test the algorithm under different illuminations. The results are better and more robust in these outdoor experiments.

4-2-3 Average Landmark Vectors

The snapshot model stores an image to represent a certain location of interest, like a nest or food source. From this, a heading vector to the home location is obtained. A similar approach is presented in (Lambrinos, Möller, Labhart, Pfeifer, & Wehner, 2000), which make use of

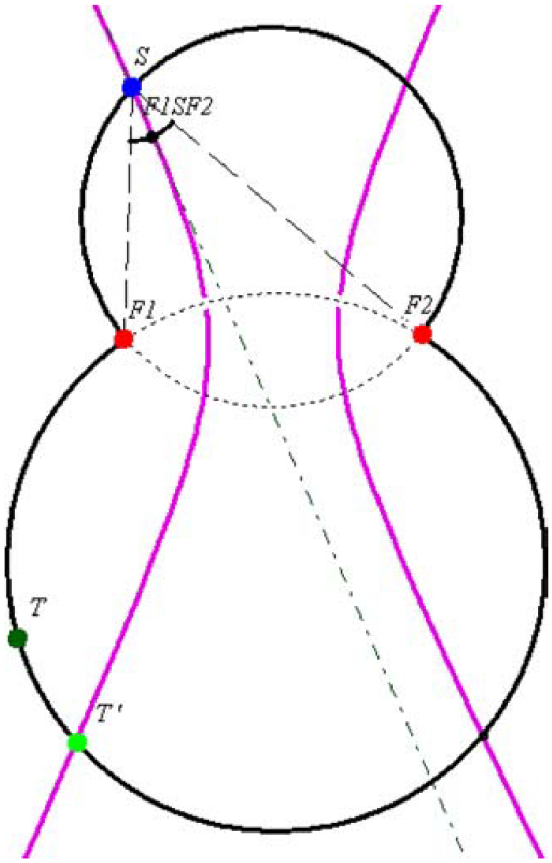


Figure 4-7: Composition of a partial motion vector from the agent location S, based on two hyperbolas with features F1 and F2 as focal points. The tangent in S with one of the hyperbolas defines the movement vector for this pair of features. Obtained from (Argyros et al., 2005).

Average Landmark Vectors (ALVs) to represent landmarks. ALVs, introduced in (Lambrinos, Möller, Pfeifer, & Wehner, 1998), are averages of the heading vectors to all landmark locations. The homing vector is determined with respect to this ALV, as shown in Figure 4-8. In (Lambrinos et al., 2000) objects are classified as landmarks using a brightness threshold on pixel intensities. When a patch of pixels above (or below) this threshold is available, it is recognized as a landmark.

The main difference between ALV homing and snapshot homing is that a location of interest is represented by a single vector, instead of an image. This both improves the computational efficiency and decreases the storage demands. The downside is that it is less accurate and prone to errors. A small offset of a landmark vector can have a big impact on the ALV and thus the homing vector. Furthermore, the current heading of a robot is needed to be able to rely on ALV.

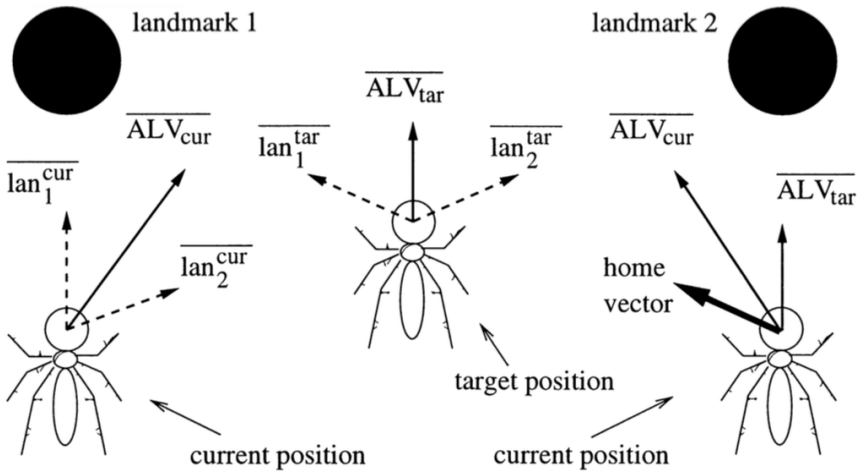


Figure 4-8: Graphic description of the ALV model, where a home vector is computed by subtracting the target ALV from the current ALV. Obtained from (Lambrinos et al., 2000).

4-2-4 Cognitive Maps

Insect-inspired navigation algorithms try to mimic the cognitive process going on in insect brains. One of the main reasons why insects are used as inspiration, is the relative simplicity of their brains, with only several hundred-thousands neurons (Mathews et al., 2009).

Next to computational limits, the simplicity of insect's brains also limit storage capabilities of route representations. In the snapshot model presented in the previous section, a retinal image is stored to identify the home or feeder location of honeybees. In more advanced landmark navigation approaches (for instance when using multiple snapshots for different locations), this route representation is already more complex, which suggests the availability of a cognitive map.

A way in which multiple retinal images can be identified as landmarks is by having a database of landmark vectors to identify the general direction of a certain food source or nest, based on the earth magnetic field (T. S. Collett & Baron, 1994).

In (Menzel et al., 2005) an experiment investigating the flight paths of bees is presented. The experiment involves kidnapping of bees when they leave a feeder location and placing them elsewhere. The results showed that bees flew straight back to a known (for instance nest) location, even when they were released far from this location. Based on these short-cuts, Menzel et al. conclude that bees must have a rich, map-like spatial memory.

Researchers are not unanimous on the usage of cognitive maps in insects like ants and bees. Observations of desert ants (which do not rely on pheromone trails) not taking the shortest route to a nest location and the observation that these ants only know the route in one direction, suggest the absence of such a map (M. Collett & Collett, 2006).

In (Cruse & Wehner, 2011) a neural model is made which makes use of PI and landmark guidance to mimic a cognitive map in an insect's brain. Based on the definition that a map allows for taking short-cuts to reach a certain landmark, they conclude that a cognitive map is not present in insects.

Both the snapshot model and ALV methods assume the presence of some cognitive map, though those maps do not have the properties of maps generated in SLAM methods (e.g., closing the loop). In the next section a method is presented where such a map is not used.

4-2-5 Scene Recognition

The use of snapshots in ant and bee navigation has recently been challenged by several researchers (Cheung et al., 2014). In (Baddeley et al., 2012) a different insect-inspired method is proposed, where pictures along the entire route are used for homing. In the paper it is posed as a likely explanation of how ants solve the homing problem (when pheromones are not effective due to the environment). Moreover, simulation results are presented which show a possible implementation on robots.

The algorithm starts as follows: one return to the nest is used to take pictures. These pictures are stored and used as familiar views. When the algorithm is applied for homing, pictures are taken with several bearings and compared against the stored frame. The *most familiar* image is used as correct direction. This is repeated for the entire route. Figure 4-9 shows the results of a homing simulation performed in (Baddeley et al., 2012). The red line indicates a single training run and the black lines show several (successful) homing experiments.

Two methods of scene storage are described in the paper. First, navigation with a perfect memory is presented, in which all pictures taken during the training run are stored. Familiarity between different images is then computed by minimizing the Sum of Squared Differences (SSD) between a stored frame and the frames taken at a location along the route.

The second storage method is the (biologically more plausible) use of an Infomax NN (Lulham, Bogacz, Vogt, & Brown, 2011), where each pixel of an image is used as input to the network. Familiarity is encoded in the weights, which are trained according to an unsupervised training scheme, as developed in (Bell & Sejnowski, 1995). A detailed description and simulation examples of the Infomax NN can be found in Chapter 5 of this thesis.

Baddeley et al. address the following three problems when performing homing using scene familiarity:

- The agent would overshoot the nest location if no stop criterion is present. The solution posed is the use of learning walks around the nest, in order to make the agent converge towards it (Muser, Sommer, Wolf, & Wehner, 2005; Müller & Wehner, 2010). An example of the usage of learning walks is shown in Figure 4-10.
- Storing multiple routes within a single NN can become a problem for network capacity. Baddeley et al. show that the algorithm still works for three different routes, although the failure rate (i.e., the amount of times the nest is not reached) is higher.
- Performance degrades when the tussock density (and hence also the number of occlusions and ambiguities) increases. This degradation is observed in the occurrence of failed returns.

A final note about the paper concerns the need for a return to the nest for training purposes. The fact that such a training run is needed, makes the method less useful for returning

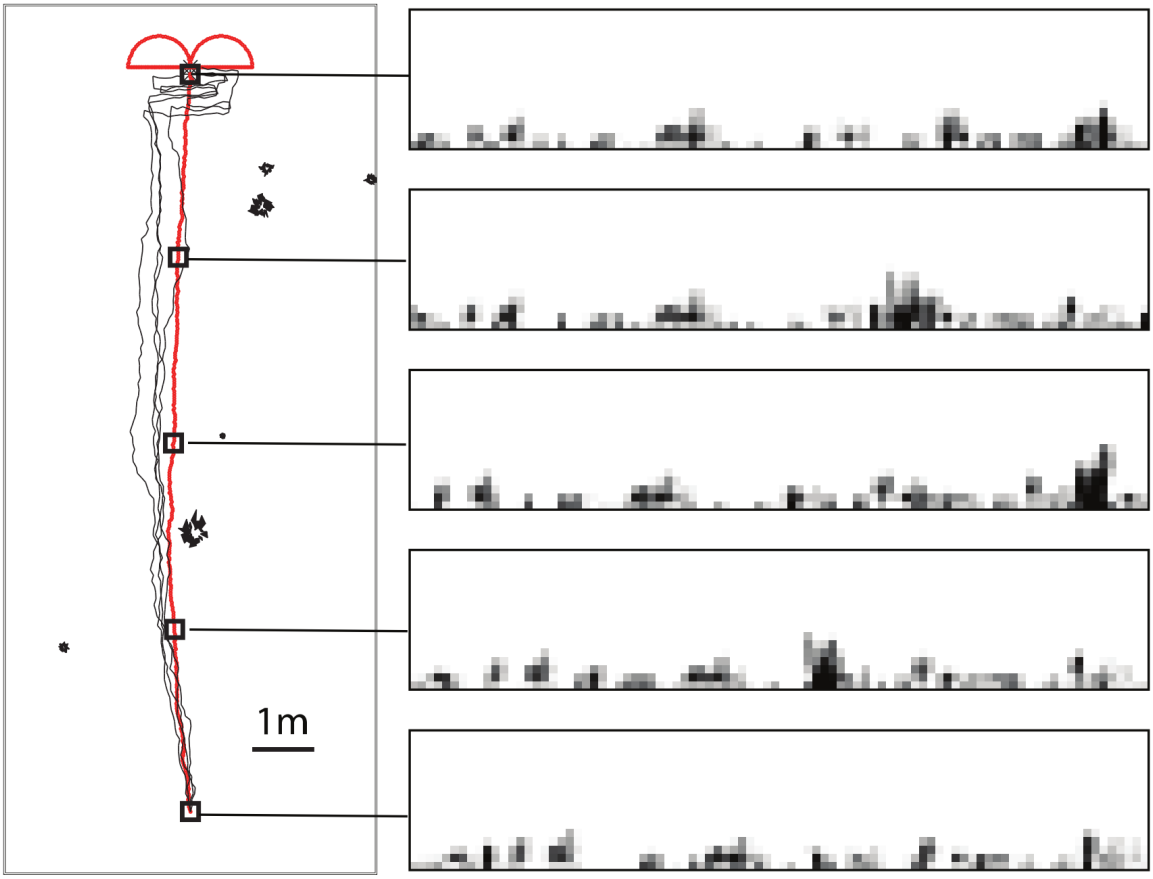


Figure 4-9: Homing results from (Baddeley et al., 2012). The left panel shows a training run (red) and test routes (black), for a route of 12 m in an environment with tussocks. The right panel shows example views from the training run.

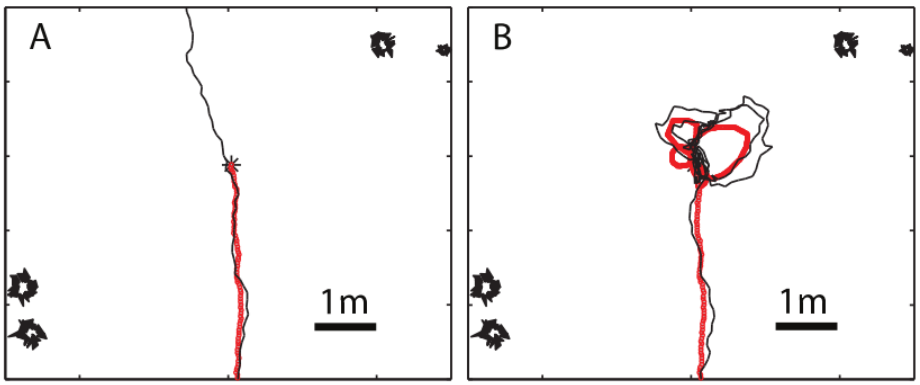


Figure 4-10: Learning walks prevent an agent to pass the nest during homing (the red lines indicate training and the black lines indicate homing simulations). Obtained from (Baddeley et al., 2012).

home after an exploratory flight. This issue could for instance be solved by using an omni-directional camera, so that training can happen during explorating. Another possibility is to adapt the flying behavior of the MAV to aid training. During this review, the usage of an omni-directional camera or implementation of special training behavior, when employing scene familiarity, was not found.

Based on the research presented in (Baddeley et al., 2012), more papers have been published by researchers from the same institute to show the superiority of the scene familiarity method over the snapshot model. In (Wystrach, Mangan, Philippides, & Graham, 2013), experiments which were originally designed to show the working of the snapshot model in navigation behavior of foragers are re-evaluated, to show that the scene familiarity approach is a more plausible model for homing behavior of insects.

Scene familiarity is a new development in insect navigation. To my knowledge, no robotic implementations using this methodology have been published yet.

Chapter 5

Infomax Neural Networks

As mentioned in the previous chapters, the approach published in (Baddeley et al., 2012) is used as basis for this thesis. The scene familiarity method makes use of an Infomax neural network (Lulham et al., 2011; Bell & Sejnowski, 1995) as representation of the route. In order to better understand the working principles and capabilities of this network, several MATLAB simulations are performed with this network.

First, Section 5-1 gives a general description of the Infomax NN for familiarity recognition and uses simple simulations to show the working of it. Then, initial research to the application of the Infomax network to vision-based scene recognition is presented in Section 5-2, using different representations of a video frame (i.e., an image). Finally, Section 5-3 lists some ideas of future research to be done in order to better understand how to robustly apply Infomax NNs on MAVs.

5-1 Infomax Neural Network Learning

As mentioned in Section 4-2-5 the scene familiarity recognition homing algorithm presented in (Baddeley et al., 2012) makes use of a neural representation to store route familiarity. This section first gives a short overview of Infomax NNs and then presents simulation results to get a better grip on how well the network is performing with regard to familiarity discrimination.

The Infomax NN used is a two-layer network with an input layer and a novelty layer (see Figure 5-1) (Lulham et al., 2011). The network can be used both for feature extraction and familiarity discrimination. However, for the method proposed in (Baddeley et al., 2012), only familiarity discrimination is needed. In both (Baddeley et al., 2012) and (Lulham et al., 2011) the number of input neurons is equal to the number of novelty neurons. In principle this is not necessary, since a lower amount of novelty neurons is computationally advantageous and might give sufficient performance for successful scene discrimination, while a higher amount should increase the storage capacity of the network (Lulham et al., 2011). In this chapter N indicates the number of inputs, while M indicates the number of novelty neurons.

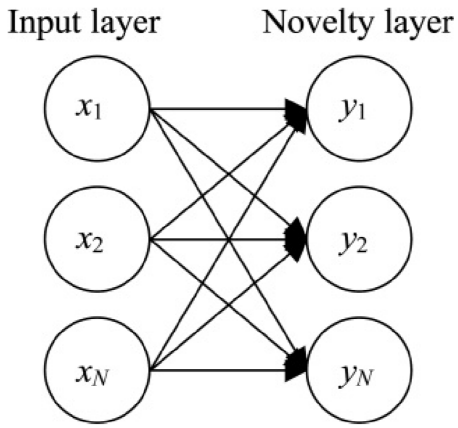


Figure 5-1: Infomax NN structure with an input layer and a novelty layer. In this representation it is assumed that the input layer and novelty layer contain an equal amount of neurons. Obtained from (Lulham et al., 2011).

The main idea behind an Infomax network for familiarity discrimination is that any sequence of inputs given as training data, will adjust the weights such that the total input to the novelty layer decreases. This metric for familiarity is defined in Equation 5-1.

$$d(x) = \sum_{i=1}^{M} |h_i| \tag{5-1}$$

Here, $d(x)$ (also called the *decision function*) is the familiarity of input sequence x, for which a smaller value means that the sequence is more familiar than when $d(x)$ is larger. h_i is the input to the ith novelty neuron, and is defined as:

$$h_i = \sum_{j=1}^{N} w_{ij} x_j \tag{5-2}$$

In this equation x_j is the input from the jth input neuron. Finally, the activation function of the ith novelty neuron is a hyperbolic tangent of h_i.

As the familiarity $d(x)$ can be seen as the desired output of this network, an output layer is discarded.

The network weights are initialized according to $\mathcal{U}(-0.5, 0.5)$, and then normalized such that the mean and standard deviation of all $\sum_{j=1}^{N} w_{ij}$ (i.e., the sum of weights for each novelty neuron) are 0 and 1 respectively.

Training is done using an unsupervised learning rule, with the aim to lower the familiarity for each given training sequence. The difference between supervised learning (which is normally used in NNs) and unsupervised learning, is that in supervised learning the difference between desired and actual output of the network is minimized to update the weights of the neurons. In unsupervised learning, however, the desired output is not used for training. Instead, an update rule as function of network input, network output, and current neuron weights is

applied to update the weights. On the one hand this makes unsupervised training very fast, but on the other hand it does not give direct control over the output of the network. As the familiarity output of an Infomax network is only used to compare the result with respect to other inputs, unsupervised learning suffices. The unsupervised learning rule used is obtained from (Lee, Girolami, & Sejnowski, 1999) and is defined as:

$$\Delta w_{i,j} = \frac{\eta}{M}\left(w_{i,j} - (y_i + h_i)\sum_{k=1}^{M} h_k w_{k,j}\right) \tag{5-3}$$

In this equation, η is the learning rate, $w_{i,j}$ is the current value of the weight between input j and neuron i, and y_i is the output of the ith novelty neuron.

To demonstrate the familiarity discrimination capabilities, a simulation is done using 100 input vectors, each containing $N = 500$ random numbers generated from $\mathcal{N}(0, 1)$. The first 50 vectors were fed to the network for training (note that the performance did not seem to vary with the number of training sequences, given a total of 100 samples). Then, the weight update rule was turned off and all 100 sequences were fed to the network to compare the familiarity. The result for a network with 200 novelty neurons and a learning rate of 10^{-3} is shown in Figure 5-2a. Even though a trend is already visible, it is hard to discriminate between familiar and novel sequences. Two ways to get a better distinction are increasing the learning rate and increasing the number of epochs (i.e., how many sequential times the weight is updated for each training sequence). Increasing the learning rate is computationally cheaper, but can lead to an unstable network. Increasing the number of epochs is more robust, but requires more computation. An example of using 10 epochs instead of 1 can be seen in Figure 5-2b. Now, it is easy to see that the first half of the sequences is familiar and used for training.

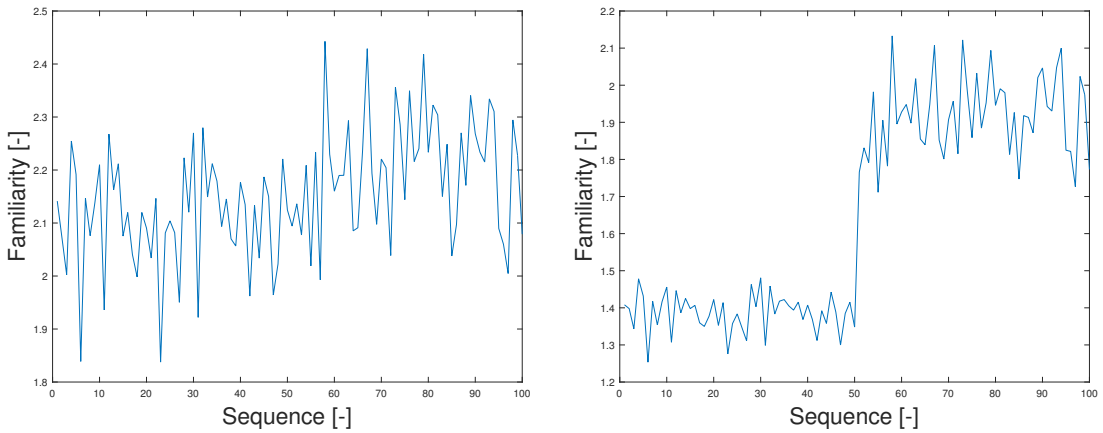


Figure 5-2: Simulation results for familiarity discrimination for 1 epoch and 10 epochs, for samples of 100 elements in $\mathcal{N}(0, 1)$.

The simulations shown give good results when a threshold is defined for the decision value (i.e., each sequence which has a decision value lower than the threshold is treated as familiar). In (Baddeley et al., 2012) no use is made of such a threshold. Instead, all measurements (i.e.,

samples) taken at different bearings are compared and the measurement with the lowest decision value is taken as the correct one.

In order to show that such recognition of the most familiar input sample works, a synthetic simulation is performed, where the height of a skyline is simulated by low-pass filtering white noise. A cutoff frequency of 0.1 rad/s is applied and the function is 1000 units wide (Figure 5-3a). One sample of 500 units is taken from this skyline (exactly in the middle) and fed to the network for training. Then, each possible view is taken from the skyline function (i.e., each possible sample with 500 units) and the familiarity d is evaluated. The result can be seen in Figure 5-3b. The figure clearly shows the sample taken in the middle of the skyline as most familiar.

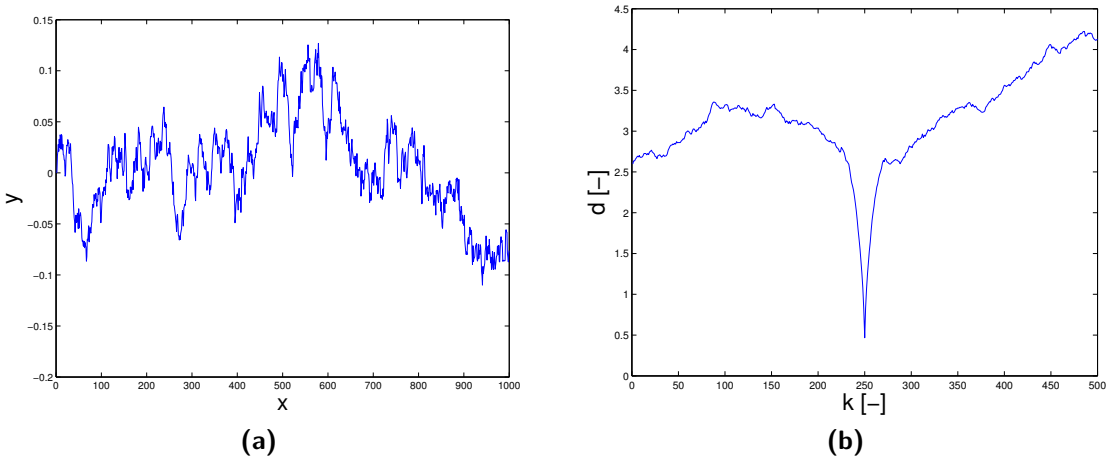


Figure 5-3: Recognition of a single trained input sample. Figure 5-3a shows low-pass filtered white noise with a cutoff frequency of 0.1 rad/s (height representation of a skyline) and Figure 5-3b shows familiarity of windows from the filtered noise of 500 units wide.

Note that an Infomax network is very sensitive to the order in which the inputs are provided to the network during training and during testing familiarity. An effective way to diminish this effect is by applying a Gaussian filter on the input sample, which effectively spreads one input over multiple, neighboring input neurons. This, however, will cause distinctive features to disappear. A trade-off is needed between sensitivity to input order and the availability of distinctive features, to find the optimal standard deviation of the Gaussian applied to the input vector.

5-2 Infomax Scene Recognition

As mentioned before, the purpose of the Infomax network in (Baddeley et al., 2012) and also in this thesis, is storage of familiar views encountered during an exploration route. The size and meaning of the network input vector should be designed such that a single sample forms a unique representation of an image. In (Baddeley et al., 2012) this is done by making a vector containing the gray scale intensities of each pixel. Simulations of this are shown in Section 5-2-1.

Using one input neuron for each pixel in an image often requires very many inputs to the network, which automatically means more novelty neurons for the same storage capacity. Instead of using pixels, different image features can be used as input to the network. In Section 5-2-2 simulations are performed with a histogram of edge features as input to the network.

In the end, the aim of the network is to be able to find a motion vector in a robust way, based on familiarity of a scene. A way to achieve this is by combining familiarity from multiple Infomax networks with different inputs. Section 5-2-3 shows simulations where a network using pixel intensities is combined with a network using edge histograms.

5-2-1 Pixel Inputs

First an Infomax network with pixel intensities as inputs is simulated. In order to replicate the results from (Baddeley et al., 2012), two simulations were set up to test the scene familiarity capabilities of the network.

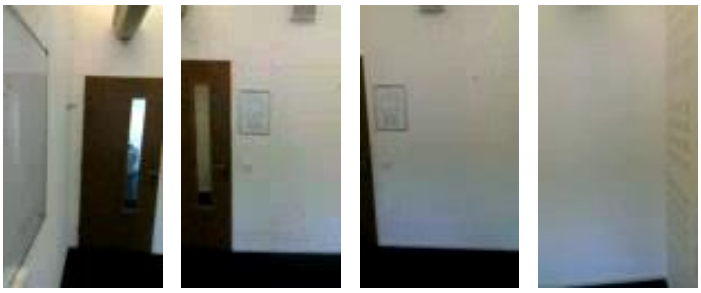
The first simulation uses a training video where an agent moves towards a wall. For testing familiarity, the agent moves parallel to the wall at a constant distance from it. The goal is for the agent to identify the view which best resembles the training set. Image sequences of four frames from the videos for training and familiarity testing are shown in Figures 5-4 and 5-5 respectively.

Figure 5-4: Four frames from the training video where an agent moves towards a wall. The video contains 173 frames in total and has a duration of 5 seconds.

The videos have a resolution of 29 by 52 pixels, which means the Infomax network has 1508 input neurons. Furthermore, the network has an (arbitrary) amount of 1000 novelty neurons and a learning rate of 0.001. Each sample is trained 20 times.

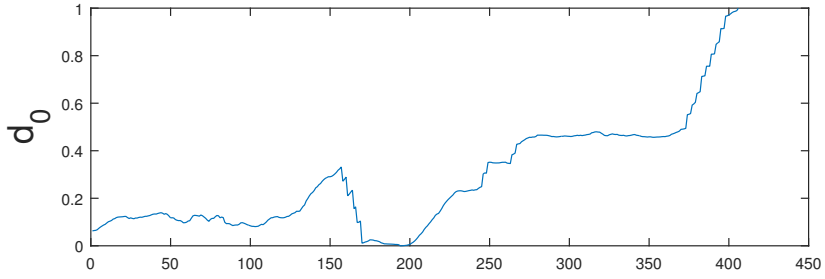
The familiarity results are shown in Figure 5-6, which contains both the familiarity before training and after training. It can be seen that the most familiar frame (i.e., it has the lowest decision value) is found to be frame 158, which corresponds to Figure 5-5b. This is indeed a good result.

The second simulation is performed in a hallway, where a walk through it is used as training video and a rotation on the spot is used for familiarity determination. Frames from the videos used for training and testing are shown in Figures 5-7 and 5-8 respectively.

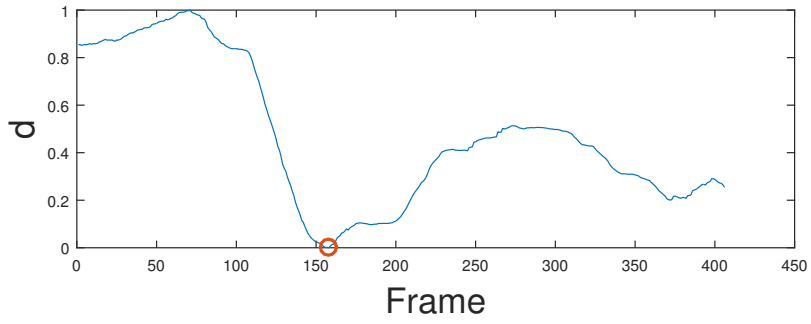


(a) (b) (c) (d)

Figure 5-5: Four frames from the testing video where an agent moves parallel to a wall. The video contains 406 frames in total and has a duration of 13 seconds.



(a)



(b)

Figure 5-6: Familiarity output of moving towards a wall experiment before and after training. The most familiar direction is found in frame 158 and is indicated with a red marker.

(a) (b) (c) (d)

Figure 5-7: Four frames from the training video where an agent moves through a hallway. The video contains 225 frames in total and has a duration of 15 seconds.

(a) (b) (c) (d)

Figure 5-8: Four frames from the testing video where an agent rotates on a spot where the direction must be determined. The video contains 91 frames in total and has a duration of 6 seconds.

The resolution of the videos is 44 by 36 pixels, which gives a total of 1584 input neurons. The number of novelty neurons, training epochs, and the learning rate are the same as in the previous experiment.

Figure 5-9 shows the familiarity results for this experiment. Frame 54 (shown in Figure 5-8c) was found as most familiar, which indeed corresponds quite well with the training video.

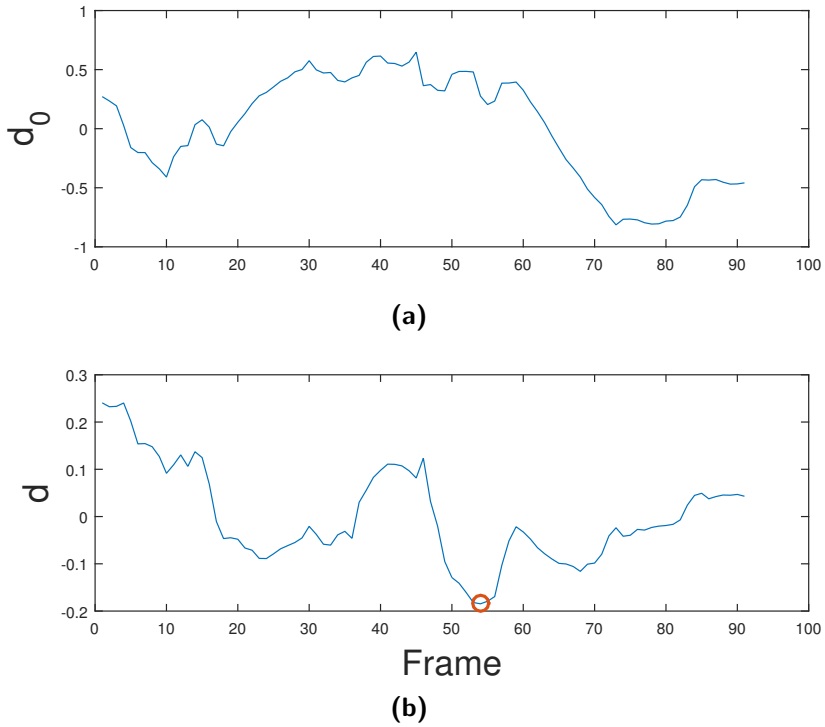


Figure 5-9: Familiarity output of walking in a hallway experiment before and after training. The most familiar direction is found in frame 54 and is indicated with a red marker.

5-2-2 Edge Inputs

As mentioned, there are other image representations which could be used as input to the Infomax NN. One possibility is the use of histograms of vertical edges.

Edges in the vertical direction are calculated using the Sobel operator and are summed over the vertical axis. This results in a one dimensional row-vector containing a histogram of edges, which can be seen as a representation of the image. Figure 5-10b shows an example of such an edge representation (to obtain a histogram, the pixel values must still be summed over the vertical axis).

(a) **(b)**

Figure 5-10: Example of vertical edges of Figure 5-5b, calculated using the Sobel operator.

Advantages of using edges are that the number of inputs to the network can be reduced and hence also the number of hidden neurons, which reduces computational demands during both training and testing. Furthermore, since only vertical edges are used, the network becomes more resistant against small differences in altitude between the training and testing videos.

To test the use of edges in the neural network, the same two experiments as described above are performed with the use of edges. For the experiment where an agent walks towards a wall, the resolution used is 540 by 960, which means that the row-vector containing the edges has 540 elements. Therefore, the network contains 540 input neurons. The number of novelty neurons (1000), training epochs (20) and the learning rate (0.001) are kept the same as before.

Figure 5-11 shows the familiarity result before and after training. The best match is again found at frame 158 (which is correct). However, the minimum decision value is less distinct than in the previous results and the familiarity after training is quite similar to the initial familiarity, which might lead to wrong solutions when this is not the case in different experiments.

The test where an agent walks through a hallway is also performed with edges as inputs. Here, the resolution of each frame is 640 by 360 pixels. Furthermore, a longer walk period of 35 seconds is used for training, which amounts to a total of 1050 frames. For testing the rotation on the spot video consists of 182 frames and has a duration of 6 seconds. The results can be seen in Figure 5-12.

The minimum decision value found is at frame 94. It can be seen however, that there is a local minimum at frame 111. In Figure 5-13 both of these frames are displayed, and it is concluded

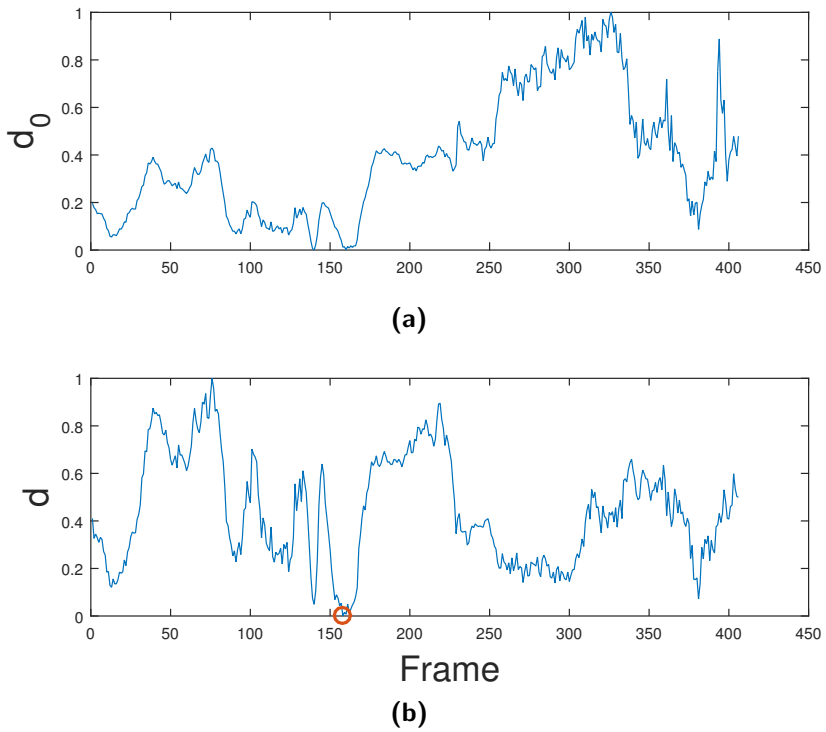


Figure 5-11: Familiarity output of moving towards a wall experiment before and after training. A histogram of vertical edges is used for familiarity. The most familiar direction is found in frame 158 and is indicated with a red marker.

that frame 111 actually is a better match to the training video. Note that Figure 5-13a is a little more blurry than 5-13b. This is the same effect as having a Gaussian filter over the inputs, which may be an explanation for the better match at the local minimum.

5-2-3 Hybrid Networks

From the tests presented in the previous sections, it can be seen that pixels used as inputs give the best results. Even so, edge histograms also give quite good results and can potentially be better when tests are performed on a flying vehicle where altitude can fluctuate.

In this section an attempt is made to improve the robustness of the results by combining both decision values in a single familiarity estimate. This is done by running two neural networks in parallel, one of which taking pixels as inputs and the other using the edge histograms. Combining the decision values is done by mapping the values of d in [0,1] (for each neural network) and adding them up. This means the familiarity solutions for both networks are weighted equally (this can be tuned in the future).

For the experiment where an agent walks towards a wall, the same videos as for the edge tests are used (i.e., with a resolution of 540 by 960 pixels). For the network with pixel intensities as inputs, this is scaled down to 34 by 60 pixels, such that 2040 input neurons are used. In order not to saturate the capacity of this network, the number of novelty neurons is increased to 2000. To increase the speed of the simulation, the number of epochs for both networks is decreased from 20 to 10.

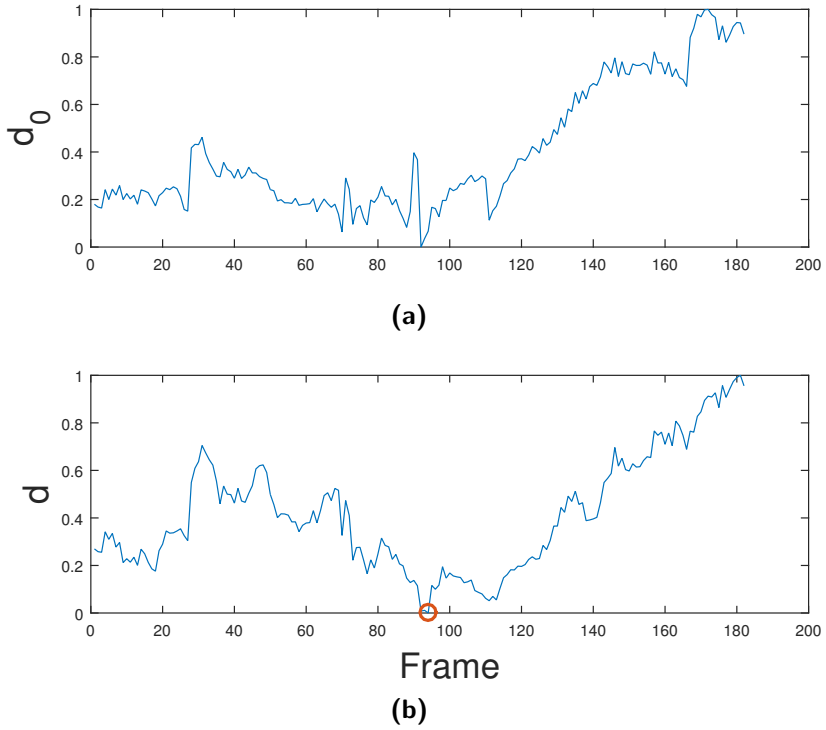


Figure 5-12: Familiarity output of walking in a hallway experiment before and after training. A histogram of vertical edges is used for familiarity. The most familiar direction is found in frame 94 and is indicated with a red marker.

Figure 5-13: Best (frame 94) and second best (frame 111) frames from the testing video of the hallway walk experiment using edge features.

The result for this experiment can be seen in Figure 5-14. Obviously, the most familiar view is still found at frame 158, but with a more distinct minimum decision value.

Finally, the hallway experiment is also performed with both pixel intensities and edge histograms as inputs. Again, the same videos as in the edge histogram experiment are used (i.e., with a resolution of 640 by 360 pixels). For the network with pixel intensities as inputs this is scaled down to 40 by 23 pixels, which give 920 inputs to this network. Both networks have 1000 novelty neurons, and 10 training epochs are used.

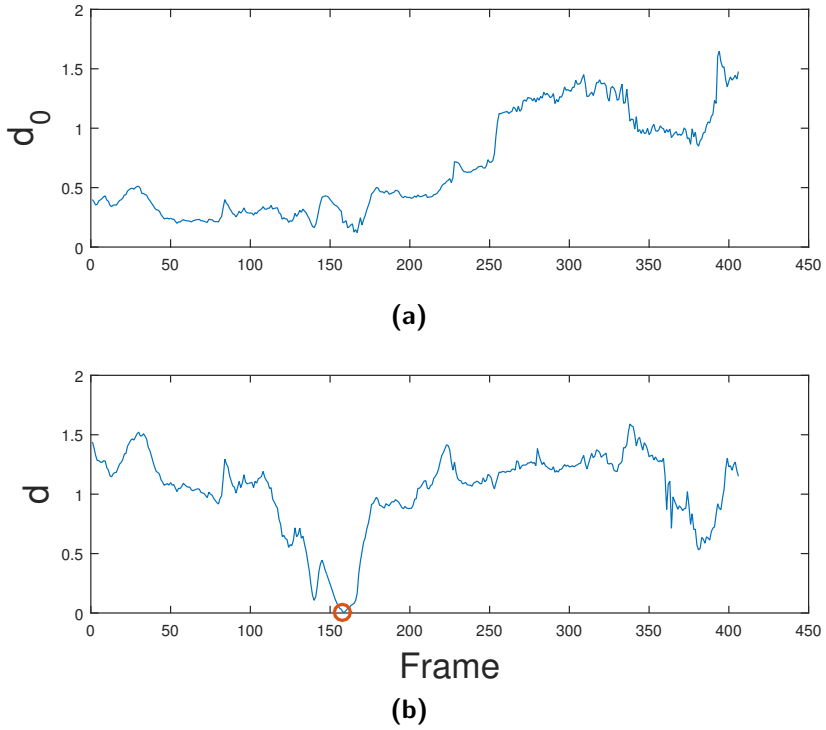


Figure 5-14: Familiarity output of moving towards a wall experiment before and after training. A combination of pixel intensities and edge histograms is used for familiarity. The most familiar direction is found in frame 158 and is indicated with a red marker.

The results of this experiment can be found in Figure 5-15. Similar to the previous combined experiment, the optimum decision value is more distinct than in the experiments with one neural network. The optimum is found at frame 107, and is close to the second best minimum (which actually was the best solution) from the experiment with only edge histograms as input.

5-3 Future Research

This chapter showed the scene recognition capabilities of an Infomax NN. Still, there are many aspects to investigate regarding the network, before implementing it on an MAV. The aspects currently found are listed here:

- Investigate whether direction can still be determined when circles are flown for training. This would make it possible to train on an exploration flight, by rotating at certain points on the trajectory and use dead-reckoning to navigate between those spots.
- Investigate the capacity of a network, both in theory (i.e., based on existing literature or synthetic simulations) and in an application on images.
- Investigate existence of other familiarity representations than Infomax.

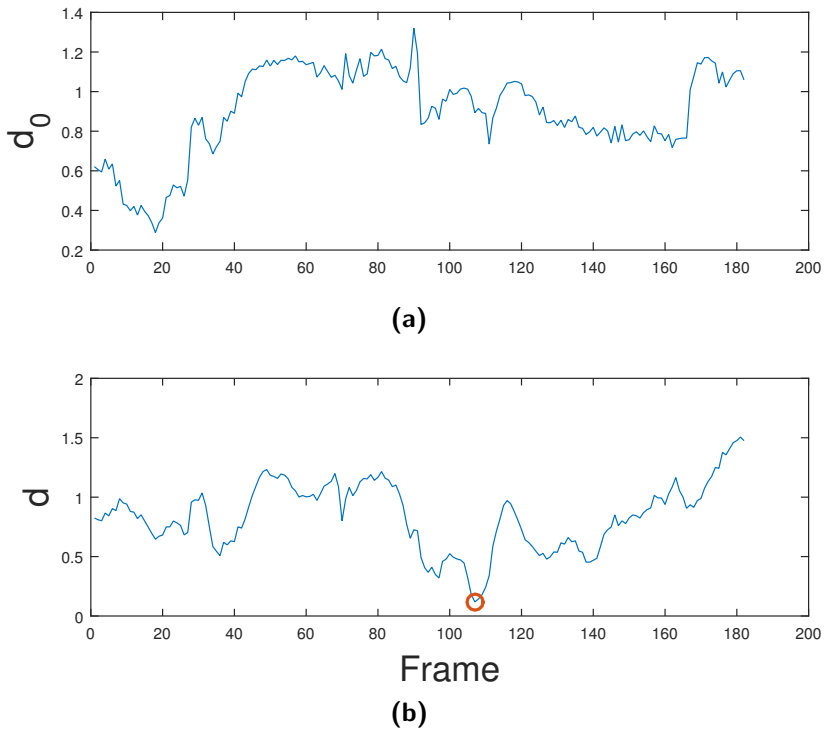


Figure 5-15: Familiarity output of walking in a hallway experiment before and after training. A combination of pixel intensities and edge histograms is used for familiarity (average familiarities of two networks). The most familiar direction is found in frame 107 and is indicated with a red marker.

- Investigate the usage of panorama images for training, instead of rotation videos. Besides the fact that this would align sequential frames during familiarity testing, a panorama is a typical output of a camera with a high field of view.
- Investigate the effects of decreasing the frame rates of videos used to train the network.
- Instead of using two networks as hybrid approach, use a single Infomax NN with both pixels and edges as inputs.

Chapter 6

SmartUAV Simulations

The cores of the papers presented in Chapters 2 and 3 consist of a familiarity analysis and closed-loop simulation results. These results are made in SmartUAV; a vision-in-the-loop simulator, developed by TU Delft. This chapter first gives a brief overview of the SmartUAV simulator, in section 6-1. Then, section 6-2 describes the image representations used in the papers in more detail, namely raw pixels, texton histograms and Hue Saturation Value (HSV) color histograms.

6-1 SmartUAV

SmartUAV is a UAV simulation platform developed in C++ at the TU Delft. It is mainly used for creating and testing UAV GNC algorithms. Due to the 3D rendered sceneries, SmartUAV is especially useful for development of computer vision algorithms for UAVs.

SmartUAV is completely modular, which means that different vehicles, sensors and algorithms can be combined easily. Different modules can be connected to eachother in a graphical manner, which enables researchers to quickly develop and test different GNC algorithms. Since the program runs on multiple threads, algorithms running at different frequencies can be used simultaneously. Simulations can be performed both in real-time and fast-time.

For the development of computer vision algorithms, the vehicles are virtually placed in a 3D environment. The fidelity of these simulated environments can be high, as needed by the researcher. To aid efficient development of computer vision methods, OpenCV is linked to SmartUAV. This makes it very easy to use well-tested computer vision algorithms.

Figure 6-1 shows a screen capture of the SmartUAV development window, where different modules can be connected to compose a GNC algorithm. Figure 6-2 shows some screen captures of a running simulation. Here, the top left figure is the Simulation Manager, where different views of the UAV can be seen by the developer. The top right window shows a trigger button, used to switch between exploration and homing. The bottom left image shows the view as observed by the UAV and the bottom middle picture contains some state values for

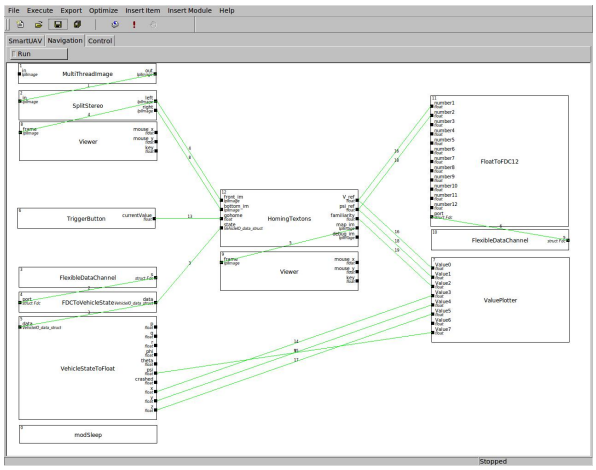


Figure 6-1: SmartUAV main window, containing the algorithm structure for homing using textons.

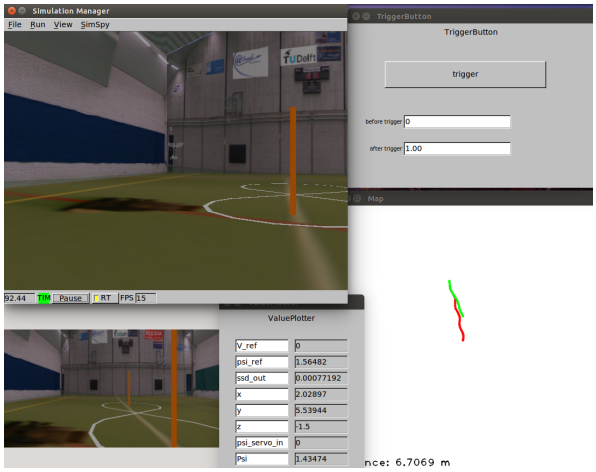


Figure 6-2: SmartUAV windows open during a homing simulation.

debugging. Finally, the bottom right window contains a top-view map, where the red line is the exploration route and the green line the homing route.

6-2 Image Representations

In the methods described in the papers, different image representations for scene familiarity are compared. These three representations are raw pixels, texton histograms and HSV color histograms, and are described in sections 6-2-1, 6-2-2 and 6-2-3 respectively.

6-2-1 Raw Pixels

As the name suggests, raw pixels represent the pixels directly coming from the camera. In the simulations performed, these are converted to grayscale and scaled down to improve real-time

performance. Figure 6-3 shows an example of an image captured in a simulated sports hall in SmartUAV. Figure 6-3a shows the raw capture from the simulated camera and Figure 6-3b shows the scaled down gray image used in simulations.

(a) (b)

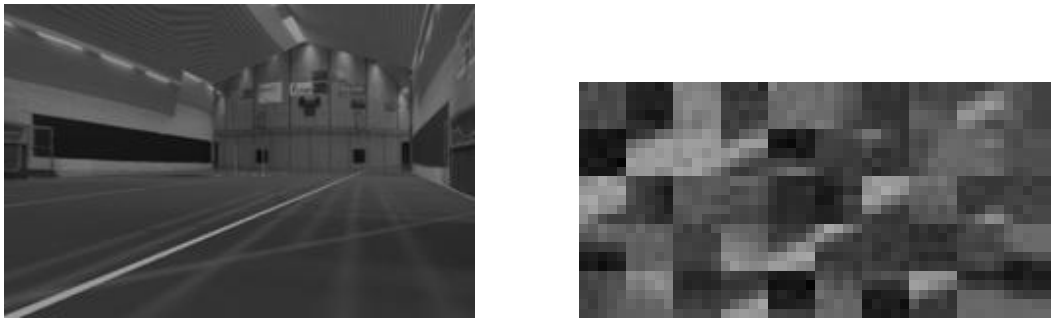
Figure 6-3: Image taken in the sports hall environment in SmartUAV. Figure 6-3a shows the raw image (480 by 320 pixels) and Figure 6-3a shows the corresponding gray, scaled down image (48 by 32 pixels).

6-2-2 Texton Histograms

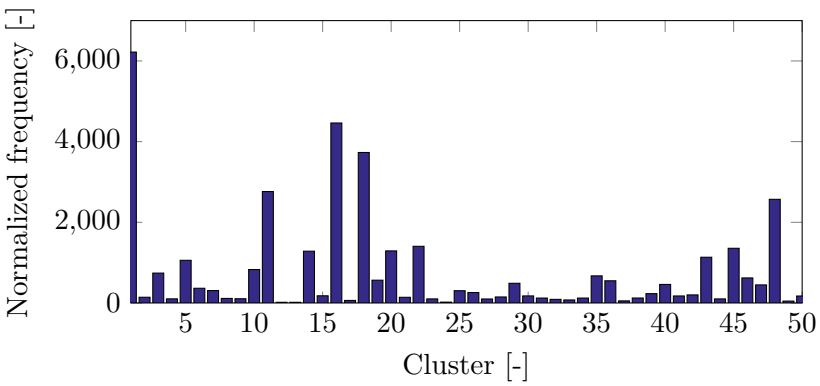
Textons are small image patches extracted from images. When classifying these patches to pre-determined clusters, a histogram can be obtained as compact representation for an image. Figure 6-4 shows an example texton representation. Figure 6-4a shows an image taken in a simulated sports hall environment, Figure 6-4b shows the set of 50 pre-determined clusters and Figure 6-4c shows a resulting texton histogram.

In the results presented in the papers, all possible patches are extracted from each image. In the case of Figure 6-4a this means that a total of 36816 image patches are extracted. Each patch is compared to each pre-determined cluster in the texton dictionary (Figure 6-4b), in order to classify it. This comparison is done by calculating the squared euclidean distance between a patch and a cluster, which is the same as computing the SSD between the two. The patch is then classified to the closest cluster. When this is done for all patches, a histogram is obtained like the one in Figure 6-4c. Note that for better computational performance, sub-sampling can be applied. This means not all possible patches are extracted from each image, but only a (random) sample.

Training of the clusters is done using the K-means clustering algorithm. For this, image patches for many (representative) images must be provided. Then, K clusters are randomly selected as initial clusters (note that $K = 50$ throughout this thesis). All other image patches in the training set are clustered to these K clusters. When this is done, for each cluster a new *center of mass* is calculated based on all patches classified in a single cluster. This center of mass is used as new cluster center, after which this process is repeated multiple times.



(a) (b)



(c)

Figure 6-4: Example of an image representation using textons. Figure 6-4a shows an example image from a sports hall. Figure 6-4b shows the clusters to which textons are assigned and Figure 6-4c shows the corresponding texton histogram. The textons are patches of 5 by 5 pixels, and a total number of 36816 textons have been extracted from the example image.

6-2-3 HSV Color Histograms

The last image representations used in this thesis are HSV color histograms. Color histograms contain a classification of each pixel based on color intensity. Here, HSV colors are used.

The three channels in the HSV color space are *hue*, *saturation* and *value*. Hue represents the color of a pixel. The value indicates an angle between 0 and 360 degrees and corresponds to a location on a colored disk. Saturation represents the amount of gray in a color; a high saturation indicates much color and little gray. Saturation is expressed as a percentage. Finally, value indicates the brightness or intensity of a pixel.

The HSV color histograms used in this paper are composed from the hue and value channels. Both are divided in 25 equal parts, which results in two histograms. For the final image representation, these histograms are concatenated.

A saturation threshold of 0.2 is used, which means all pixels with a saturation lower than this value are discarded in the histograms. This is to make sure that only illuminated pixels are used for the representation.

Bibliography

Angeli, A., Doncieux, S., Meyer, J.-A., & Filliat, D. (2008a, sep). Incremental vision-based topological SLAM. In *2008 IEEE/RSJ international conference on intelligent robots and systems.* New York, USA: IEEE. doi: 10.1109/iros.2008.4650675

Angeli, A., Doncieux, S., Meyer, J.-A., & Filliat, D. (2008b, may). Real-time visual loop-closure detection. In *2008 IEEE international conference on robotics and automation.* New York, USA: IEEE. doi: 10.1109/robot.2008.4543475

Argyros, A. A., Bekris, K. E., Orphanoudakis, S. C., & Kavraki, L. E. (2005, jul). Robot homing by exploiting panoramic vision. *Autonomous Robots*, *19*(1), 7–25. doi: 10.1007/s10514-005-0603-7

Baddeley, B., Graham, P., Husbands, P., & Philippides, A. (2012, jan). A model of ant route navigation driven by scene familiarity. *PLoS Computational Biology*, *8*(1). doi: 10.1371/journal.pcbi.1002336

Barron, J. L., Fleet, D. J., & Beauchemin, S. S. (1994, feb). Performance of optical flow techniques. *International Journal of Computer Vision*, *12*(1), 43–77. doi: 10.1007/bf01420984

Bell, A. J., & Sejnowski, T. J. (1995, nov). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, *7*(6), 1129–1159. doi: 10.1162/neco.1995.7.6.1129

Blosch, M., Weiss, S., Scaramuzza, D., & Siegwart, R. (2010, may). Vision based MAV navigation in unknown and unstructured environments. In *2010 IEEE international conference on robotics and automation.* New York, USA: IEEE. doi: 10.1109/robot.2010.5509920

Bonin-Font, F., Ortiz, A., & Oliver, G. (2008, may). Visual navigation for mobile robots: A survey. *Journal of Intelligent Robotic Systems*, *53*(3), 263–296. doi: 10.1007/s10846-008-9235-4

Caetano, J. V., de Visser, C. C., de Croon, G. C. H. E., Remes, B., de Wagter, C., Verboom, J., & Mulder, M. (2013, dec). Linear aerodynamic model identification of a flapping wing MAV based on flight test data. *International Journal of Micro Air Vehicles*, *5*(4), 273–286. doi: 10.1260/1756-8293.5.4.273

Cartwright, B. A., & Collett, T. S. (1983). Landmark learning in bees. *Journal of Comparative Physiology*, *151*(4), 521–543. doi: 10.1007/bf00605469

Cartwright, B. A., & Collett, T. S. (1987, aug). Landmark maps for honeybees. *Biological Cybernetics*, *57*(1-2), 85–93. doi: 10.1007/bf00318718

Chao, H., Gu, Y., & Napolitano, M. (2013, oct). A survey of optical flow techniques for robotics navigation applications. *Journal of Intelligent & Robotic Systems*, *73*(1-4), 361–372. doi: 10.1007/s10846-013-9923-6

Cheung, A., Collett, M., Collett, T. S., Dewar, A., Dyer, F., Graham, P., ... Zeil, J. (2014, oct). Still no convincing evidence for cognitive map use by honeybees. *Proceedings of the National Academy of Sciences*, *111*(42), E4396–E4397. doi: 10.1073/pnas.1413581111

Choset, H. (2000, feb). Sensor-based exploration: Incremental construction of the hierarchical generalized voronoi graph. *The International Journal of Robotics Research*, *19*(2), 126–148. doi: 10.1177/02783640022066789

Clemente, L., Davison, A., Reid, I., Neira, J., & Tards, J. (2007, June). Mapping large loops with a single hand-held camera. In *Proceedings of robotics: Science and systems.* Atlanta, GA, USA: MIT Press.

Collett, M., & Collett, T. S. (2006, jan). Insect navigation: No map at the end of the trail? *Current Biology*, *16*(2), R48–R51. doi: 10.1016/j.cub.2006.01.007

Collett, T. S. (1996). Insect navigation en route to the goal: Multiple strategies for the use of landmarks. *Journal of Experimental Biology*, *199*, 227–235.

Collett, T. S., & Baron, J. (1994, mar). Biological compasses and the coordinate frame of landmark memories in honeybees. *Nature*, *368*(6467), 137–140. doi: 10.1038/368137a0

Collett, T. S., & Collett, M. (2002, jul). Memory use in insect visual navigation. *Nature Reviews Neuroscience*, *3*(7), 542–552. doi: 10.1038/nrn872

Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C., et al. (2001). *Introduction to algorithms* (Vol. 2). Cambridge, MA, USA: MIT Press.

Cruse, H., & Wehner, R. (2011, mar). No need for a cognitive map: Decentralized memory for insect navigation. *PLoS Computational Biology*, *7*(3). doi: 10.1371/journal.pcbi.1002009

de Croon, G., de Clercq, K., Ruijsink, R., Remes, B., & de Wagter, C. (2009, jun). Design, aerodynamics, and vision-based control of the DelFly. *International Journal of Micro Air Vehicles*, *1*(2), 71–97. doi: 10.1260/175682909789498288

de Croon, G. C. H. E., Groen, M. A., de Wagter, C., Remes, B., Ruijsink, R., & van Oudheusden, B. W. (2012, may). Design, aerodynamics and autonomy of the DelFly. *Bioinspiration & Biomimetics*, *7*(2). doi: 10.1088/1748-3182/7/2/025003

de Croon, G. C. H. E., Ho, H. W., de Wagter, C., van Kampen, E., Remes, B., & Chu, Q. P. (2013, dec). Optic-flow based slope estimation for autonomous landing. *International Journal of Micro Air Vehicles*, *5*(4), 287–298. doi: 10.1260/1756-8293.5.4.287

de Wagter, C., Tijmons, S., Remes, B. D. W., & de Croon, G. C. H. E. (2014, may). Autonomous flight of a 20-gram flapping wing MAV with a 4-gram onboard stereo vision system. In *2014 IEEE international conference on robotics and automation (ICRA).* New York, USA: IEEE. doi: 10.1109/icra.2014.6907589

Desouza, G., & Kak, A. (2002). Vision for mobile robot navigation: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(2), 237–267. doi: 10.1109/34.982903

Diamantas, S. C., Oikonomidis, A., & Crowder, R. M. (2010, jul). Towards optical flow-based robotic homing. In *The 2010 international joint conference on neural networks*

(IJCNN). New York, USA: IEEE. doi: 10.1109/ijcnn.2010.5596621

Ding, W., Wang, J., Han, S., Almagbile, A., Garratt, M. A., Lambert, A., & Wang, J. J. (2009). Adding Optical Flow into the GPS/INS Integration for UAV navigation. In *Proceedings of International Global Navigation Satellite Systems Society Symposium* (pp. 1–13).

Dissanayake, M., Newman, P., Clark, S., Durrant-Whyte, H., & Csorba, M. (2001, jun). A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, *17*(3), 229–241. doi: 10.1109/70.938381

Durrant-Whyte, H., & Bailey, T. (2006, jun). Simultaneous localization and mapping: part i. *IEEE Robotics & Automation Magazine*, *13*(2), 99–110. doi: 10.1109/mra.2006.1638022

Eade, E., & Drummond, T. (2008). Unified loop closing and recovery for real time monocular SLAM. In *Procedings of the british machine vision conference 2008.* Durham, UK: British Machine Vision Association. doi: 10.5244/c.22.6

Esch, H., & Burns, J. (1996). Distance estimation by foraging honeybees. *The Journal of experimental biology*, *199*(1), 155–162.

Forster, C., Pizzoli, M., & Scaramuzza, D. (2014, may). SVO: Fast semi-direct monocular visual odometry. In *2014 IEEE international conference on robotics and automation (ICRA)*. New York, USA: IEEE. doi: 10.1109/icra.2014.6906584

Franz, M. O., & Mallot, H. A. (2000, jan). Biomimetic robot navigation. *Robotics and Autonomous Systems*, *30*(1-2), 133–153. doi: 10.1016/s0921-8890(99)00069-x

Franz, M. O., Schölkopf, B., Mallot, H. A., & Bülthoff, H. H. (1998, oct). Where did i take that snapshot? scene-based homing by image matching. *Biological Cybernetics*, *79*(3), 191–202. doi: 10.1007/s004220050470

Fraundorfer, F., Engels, C., & Nister, D. (2007, oct). Topological mapping, localization and navigation using image collections. In *2007 IEEE/RSJ international conference on intelligent robots and systems.* New York, USA: IEEE. doi: 10.1109/iros.2007.4399123

Frese, U., Larsson, P., & Duckett, T. (2005, apr). A multilevel relaxation algorithm for simultaneous localization and mapping. *IEEE Transactions on Robotics*, *21*(2), 196–207. doi: 10.1109/tro.2004.839220

Fuentes-Pacheco, J., Ruiz-Ascencio, J., & Rendón-Mancha, J. M. (2012, nov). Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review*, *43*(1), 55–81. doi: 10.1007/s10462-012-9365-8

Gallistel, C. R. (1990). *The Organization of Learning.* Cambridge, MA, USA: MIT Press.

Garcia-Fidalgo, E., & Ortiz, A. (2015, feb). Vision-based topological mapping and localization methods: A survey. *Robotics and Autonomous Systems*, *64*, 1–20. doi: 10.1016/j.robot.2014.11.009

Gil, A., Reinoso, Ó., Ballesta, M., & Juliá, M. (2010, jan). Multi-robot visual SLAM using a rao-blackwellized particle filter. *Robotics and Autonomous Systems*, *58*(1), 68–80. doi: 10.1016/j.robot.2009.07.026

Graham, P. (2010). Insect navigation. In *Encyclopedia of animal behavior* (pp. 167–175). Amsterdam: Elsevier. doi: 10.1016/b978-0-08-045337-8.00067-x

Haralick, B. M., Lee, C.-N., Ottenberg, K., & Nölle, M. (1994, dec). Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal of Computer Vision*, *13*(3), 331–356. doi: 10.1007/bf02028352

Harris, C., & Stephens, M. (1988). A combined corner and edge detector. In *Procedings of the alvey vision conference 1988.* Manchester, UK: Alvey Vision Club. doi: 10.5244/c.2.23

Ho, K. L., & Newman, P. (2006, sep). Loop closure detection in SLAM by combining visual and spatial appearance. *Robotics and Autonomous Systems*, *54*(9), 740–749. doi: 10.1016/j.robot.2006.04.016

Ho, K. L., & Newman, P. (2007, jan). Detecting loop closure with scene sequences. *International Journal of Computer Vision*, *74*(3), 261–286. doi: 10.1007/s11263-006-0020-1

Horn, B. K., & Schunck, B. G. (1981, nov). Determining optical flow. In J. J. Pearson (Ed.), *Techniques and applications of image understanding.* SPIE. doi: 10.1117/12.965761

Klein, G., & Murray, D. (2007, nov). Parallel tracking and mapping for small AR workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality.* New York, USA: IEEE. doi: 10.1109/ismar.2007.4538852

Kuipers, B., & Byun, Y.-T. (1991, nov). A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robotics and Autonomous Systems*, *8*(1-2), 47–63. doi: 10.1016/0921-8890(91)90014-c

Lambrinos, D., Möller, R., Labhart, T., Pfeifer, R., & Wehner, R. (2000, jan). A mobile robot employing insect strategies for navigation. *Robotics and Autonomous Systems*, *30*(1-2), 39–64. doi: 10.1016/s0921-8890(99)00064-0

Lambrinos, D., Möller, R., Pfeifer, R., & Wehner, R. (1998). Landmark navigation without snapshots: the average landmark vector model. In *Proceedings of neurobiology conference göttingen.*

Lamon, P., Nourbakhsh, I., Jensen, B., & Siegwart, R. (2001). Deriving and matching image fingerprint sequences for mobile robot localization. In *Proceedings 2001 ICRA. IEEE international conference on robotics and automation.* New York, USA: IEEE. doi: 10.1109/robot.2001.932841

Lee, T.-W., Girolami, M., & Sejnowski, T. J. (1999, feb). Independent component analysis using an extended infomax algorithm for mixed subgaussian and supergaussian sources. *Neural Computation*, *11*(2), 417–441. doi: 10.1162/089976699300016719

Leonard, J., & Durrant-Whyte, H. (1991). Simultaneous map building and localization for an autonomous mobile robot. In *Proceedings IROS '91:IEEE/RSJ international workshop on intelligent robots and systems '91.* New York, USA: IEEE. doi: 10.1109/iros.1991.174711

Levitt, T. S., & Lawton, D. T. (1990, aug). Qualitative navigation for mobile robots. *Artificial Intelligence*, *44*(3), 305–360. doi: 10.1016/0004-3702(90)90027-w

Lowe, D. G. (2004, nov). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*(2), 91–110. doi: 10.1023/b:visi.0000029664.99615.94

Lucas, B. D., & Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2* (pp. 674–679). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Lulham, A., Bogacz, R., Vogt, S., & Brown, M. W. (2011, apr). An infomax algorithm can perform both familiarity discrimination and feature extraction in a single network. *Neural Computation*, *23*(4), 909–926. doi: 10.1162/neco_a_00097

Magree, D. P., & Johnson, E. N. (2015, jan). A monocular vision-aided inertial navigation system with improved numerical stability. In *AIAA guidance, navigation, and control conference.* Reston, VA, USA: AIAA. doi: 10.2514/6.2015-0097

Magree, D. P., van Dalen, G. J. J., Haviland, S., & Johnson, E. N. (2015). Light-weight quadrotor with on-board absolute vision-aided navigation. In *2015 International Con-*

ference on Unmanned Aircraft Systems (ICUAS). New York, USA: IEEE.

Mathews, Z., Lechon, M., Calvo, J. B., Dhir, A., Duff, A., i Badia, S. B., & Verschure, P. F. (2009, oct). Insect-like mapless navigation based on head direction cells and contextual learning using chemo-visual sensors. In *2009 IEEE/RSJ international conference on intelligent robots and systems.* New York, USA: IEEE. doi: 10.1109/iros.2009.5354264

Menzel, R., Greggers, U., Smith, A., Berger, S., Brandt, R., Brunke, S., ... Watzl, S. (2005, feb). Honey bees navigate according to a map-like spatial memory. *Proceedings of the National Academy of Sciences*, *102*(8), 3040–3045. doi: 10.1073/pnas.0408550102

Milford, M., Wyeth, G., & Prasser, D. (2004). RatSLAM: a hippocampal model for simultaneous localization and mapping. In *IEEE international conference on robotics and automation, 2004. proceedings. ICRA '04. 2004.* New York, USA: IEEE. doi: 10.1109/robot.2004.1307183

Möller, R., Maris, M., & Lambrinos, D. (1999, jun). A neural model of landmark navigation in insects. *Neurocomputing*, *26-27*, 801–808. doi: 10.1016/s0925-2312(98)00150-7

Motard, E., Raducanu, B., Cadenat, V., & Vitria, J. (2007, apr). Incremental on-line topological map learning for a visual homing application. In *Proceedings 2007 IEEE international conference on robotics and automation.* New York, USA: IEEE. doi: 10.1109/robot.2007.363623

Muller, M., & Wehner, R. (1988, jul). Path integration in desert ants, cataglyphis fortis. *Proceedings of the National Academy of Sciences*, *85*(14), 5287–5290. doi: 10.1073/pnas.85.14.5287

Müller, M., & Wehner, R. (2010, aug). Path integration provides a scaffold for landmark learning in desert ants. *Current Biology*, *20*(15), 1368–1371. doi: 10.1016/j.cub.2010.06.035

Muser, B., Sommer, S., Wolf, H., & Wehner, R. (2005). Foraging ecology of the thermophilic australian desert ant, melophorus bagoti foraging ecology of the thermophilic australian desert ant, melophorus bagoti. *Australian Journal of Zoology*, *53*(5), 301. doi: 10.1071/zo05023

Nelson, R. C. (1991, aug). Visual homing using an associative memory. *Biological Cybernetics*, *65*(4), 281–291. doi: 10.1007/bf00206225

Newman, P., Leonard, J., Tardos, J., & Neira, J. (2002). Explore and return: experimental validation of real-time concurrent mapping and localization. In *Proceedings 2002 IEEE international conference on robotics and automation.* New York, USA: IEEE. doi: 10.1109/robot.2002.1014803

Nistér, D. (2005, nov). Preemptive RANSAC for live structure and motion estimation. *Machine Vision and Applications*, *16*(5), 321–329. doi: 10.1007/s00138-005-0006-y

Nister, D., Naroditsky, O., & Bergen, J. (2004). Visual odometry. In *Proceedings of the 2004 IEEE computer society conference on computer vision and pattern recognition.* New York, USA: IEEE. doi: 10.1109/cvpr.2004.1315094

Olson, E., Leonard, J., & Teller, S. (2006). Fast iterative alignment of pose graphs with poor initial estimates. In *Proceedings 2006 IEEE international conference on robotics and automation, 2006. ICRA 2006.* New York, USA: IEEE. doi: 10.1109/robot.2006.1642040

Pons, J. S., Hübner, W., Dahmen, H., & Mallot, H. (2007). Vision-based robot homing in dynamic environments. In *13th IASTED International Conference on Robotics and Applications* (pp. 293–298).

Shi, J., & Tomasi. (1994). Good features to track. In *Proceedings of IEEE conference on computer vision and pattern recognition CVPR-94.* New York, USA: IEEE Computer Society Press. doi: 10.1109/cvpr.1994.323794

Smith, R., Self, M., & Cheeseman, P. (1990). Estimating uncertain spatial relationships in robotics. In *Autonomous robot vehicles* (pp. 167–193). New York, USA: Springer. doi: 10.1007/978-1-4613-8997-2_14

Srinivasan, M. V. (1994, sep). An image-interpolation technique for the computation of optic flow and egomotion. *Biological Cybernetics*, *71*(5), 401–415. doi: 10.1007/bf00198917

Srinivasan, M. V. (2014, apr). Going with the flow: a brief history of the study of the honeybee's navigational 'odometer'. *Journal of Comparitive Physiology A*, *200*(6), 563–573. doi: 10.1007/s00359-014-0902-6

Srinivasan, M. V., Zhang, S., & Bidwell, N. (1997). Visually mediated odometry in honeybees. *The Journal of Experimental Biology*, *200*(19), 2513–2522.

Srinivasan, M. V., Zhang, S. W., Lehrer, M., & Collett, T. S. (1996, jan). Honeybee navigation en route to the goal: visual flight control and odometry. *Journal of Experimental Biology*, *199*, 237–244.

Strydom, R., Thurrowgood, S., & Srinivasan, M. V. (2014). Visual odometry: autonomous uav navigation using optic flow and stereo. In *Australasian Conference on Robotics and Automation (ACRA)(Melbourne).*

Tapus, A. (2005). *Topological SLAM - Simultaneous Localization and Mapping with Fingerprints of Places* (Unpublished doctoral dissertation). École Polytechnique Fédérale de Lausanne.

Tapus, A., & Siegwart, R. (2005). Incremental robot mapping with fingerprints of places. In *2005 IEEE/RSJ international conference on intelligent robots and systems.* New York, USA: IEEE. doi: 10.1109/iros.2005.1544977

Thrun, S. (1998, feb). Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, *99*(1), 21–71. doi: 10.1016/s0004-3702(97)00078-7

Thrun, S., & Bü, A. (1996). Integrating grid-based and topological maps for mobile robot navigation. In *Proceedings of the thirteenth national conference on artificial intelligence - volume 2* (pp. 944–950). AAAI Press.

Tuytelaars, T., & Mikolajczyk, K. (2007). Local invariant feature detectors: A survey. *FNT in Computer Graphics and Vision*, *3*(3), 177–280. doi: 10.1561/0600000017

Vardy, A., & Moller, R. (2005, mar). Biologically plausible visual homing methods based on optical flow techniques. *Connection Science*, *17*(1-2), 47–89. doi: 10.1080/09540090500140958

Vidal-Calleja, T. A., Berger, C., Solà, J., & Lacroix, S. (2011, sep). Large scale multiple robot visual mapping with heterogeneous landmarks in semi-structured terrain. *Robotics and Autonomous Systems*, *59*(9), 654–674. doi: 10.1016/j.robot.2011.05.008

Weiss, S., Achtelik, M. W., Lynen, S., Achtelik, M. C., Kneip, L., Chli, M., & Siegwart, R. (2013, aug). Monocular vision for long-term micro aerial vehicle state estimation: A compendium. *Journal of Field Robotics*, *30*(5), 803–831. doi: 10.1002/rob.21466

Williams, B., Klein, G., & Reid, I. (2011, sep). Automatic relocalization and loop closing for real-time monocular SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *33*(9), 1699–1712. doi: 10.1109/tpami.2011.41

Williams, B. P. (2009). *Simultaneous localisation and mapping using a single camera* (Unpublished doctoral dissertation). Oxford University.

Wolf, H. (2011, apr). Odometry and insect navigation. *Journal of Experimental Biology*, *214*(10), 1629–1641. doi: 10.1242/jeb.038570

Wystrach, A., Mangan, M., Philippides, A., & Graham, P. (2013, jan). Snapshots in ants? new interpretations of paradigmatic experiments. *Journal of Experimental Biology*, *216*(10), 1766–1770. doi: 10.1242/jeb.082941