Hindernisvermeidung bei Wüstenameisen
Entwurf und Testen eines virtuellen Agenten

Diplomarbeit
der Fakultät für Biologie
der Eberhard-Karls-Universität Tübingen

vorgelegt von
Schuchmann Roy
Tübingen, Juli 2006

Erklärung:
Hiermit erkläre ich, dass ich diese Arbeit selbst verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Tübingen, den 11.07.06

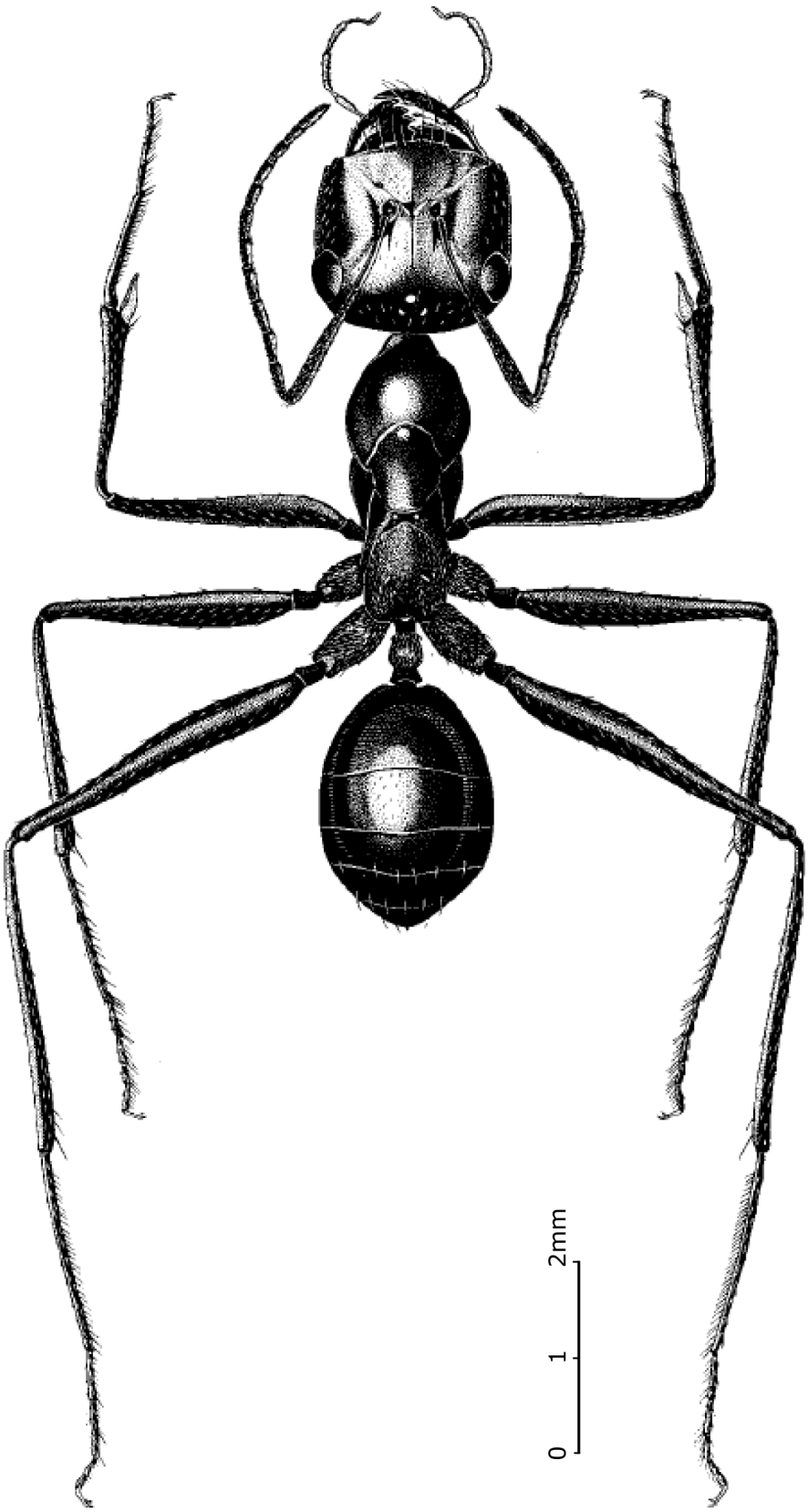
Obstacle Avoidance in Desert Ants
Design and Test of a Virtual Agent

Diploma Thesis
of the Faculty of Biology
Eberhard-Karls-University, Tübingen
presented by

Schuchmann Roy

Under supervision of
Prof. Dr. H. A. Mallot
and
Dipl. Biol. Kai Basten
Department of Cognitive Neuroscience
Eberhard-Karls-Universität Tübingen

August 11, 2006


0
1
2mm

On page 3, *Cataglyphis fortis*, dorsal view. Source [23]

1 Zusammenfassung in deutscher Sprache

Eine gut gezielte Bewegung durch den Raum ist eine wichtige Fähigkeit im Leben der meisten Tiere. Bei der Suche nach Nahrung, einem passenden Partner oder einem Lebensraum müssen Tiere in der Lage sein, Hindernisse, die ihren gewünschten Weg blockieren, zu vermeiden, wenn sie einem erfolgreichen Kurs im Leben folgen wollen. Für einige Organismen, die ein Nest wie Vögel, Insekten oder Säugetiere als ihre zentrale Lebensstelle verwenden, ist die Fähigkeit, Wissen über ihre Position im Raum zu behalten, zwingend erforderlich. Diese Fähigkeit hilft solchen Arten, eine hoch komplexe Verhaltensaktivität zu erzielen. Seit Anfang des 20. Jahrhunderts ist es Forschern bekannt, dass Ameisen auf dem kürzesten Weg zu ihrem Nest navigieren können, nach dem sie eine Futterquelle gefunden haben. Um diese Navigationsaufgabe korrekt ausführen zu können, verwenden Ameisen zwei "Werkzeuge", die ihnen zur Verfügung stehen: einen Sonnenkompass und einen Schrittzähler. Mit Hilfe des Sonnenkompasses kennt die Ameise ihre aktuelle Position und durch die Verwendung des Schrittzählers kann sie ihre gelaufene Distanz vom Ausgangspunkt messen. Mit Hilfe einer einfachen Triangulation ist die Ameise dann immer in der Lage, ihre Position relativ zum Ziel zu berechnen und so zu jedem beliebigen Zeitpunkt zum Ausgangspunkt zurück zu gelangen. Diese Studie strebt an, das Verhalten der Wüstenameisen beim Antreffen von Hindernissen auf dem Weg zu ihrem Nest oder zu einer Futterstelle zu modellieren. Dies wird durch ein virtuelles Modell der Wegintegration und der Hindernisvermeidung, basierend auf einen Echtzeit visuellen Input, erreicht. Beide Modelle werden hier im Computer entwickelt und die Integration beider Modelle miteinander wird geprüft. Die Arbeit, die hier dargestellt wird, basiert auf wissenschaftlichen Daten aus dem Bereich der Navigation und Hindernisvermeidung bei Wüstenameisen und Robotern. Das Wissen, das in einer Anzahl von Experimenten gesammelt wurde, wurde verwendet, um eine dreidimensionale (3D) Computersimulation zu konstruieren, die versucht, das Laufverhalten der Wüstenameisen wiederzugeben. Diese Arbeit bietet einige mögliche Lösungen in Form von Algorithmen an, die durch Wüstenameisen für das Lösen der Aufgabe von Hindernisvermeidung gebräuchlich sein könnte. Die 3D-Umgebung, die in dieser Arbeit verwendet wurde, wurde aus einer Region im zentralen australischen Weideland, 17 Kilometer westlich von Alice Springs, rekonstruiert. Diese Region ist auch der natürliche Lebensraum der australischen Wüstenameisen *Melophorus bagoti*, die als die rote Honigameise bekannt ist. Diese Region wird durch die Grasbüsche Cenchrus ciliaris abgedeckt, die willkürlich in der Umgebung verteilt sind. Die Büsche sind ungefähr 10-60 Zentimeter groß und der Boden ist eine sandige und ziemlich flache Oberfläche aus Lehm. Weitere Vegetation in der Umgebung waren weit zerstreute Bäume von *Akazie estrophiolata* und von *Hakea eyreana*. In dieser Arbeit wurde auch Literatur von Versuchen verwendet, die sich mit dem Laufverhalten der Ameise *Cataglyphis fortis* beschäftigen. Diese Ameise lebt in der Tunesischen Wüste und wurde auch dort untersucht. Das Laufverhalten der virtuellen Ameise wurde durch zwei Faktoren beeinflusst: erstens durch die Richtung des Ziels in dessen Richtung die Ameise laufen wollte (meistens die Futterstelle) und zweitens durch Hindernisse die den direkten Weg der Ameise zum Ziel blockiert hatten. Das visuelle Feld der Ameise (beträgt fast 360^o) ist in Segmente geteilt und die Anzahl der Hindernispixel in jedem Segment wird separat gezählt. Anhand der Anzahl der Hindernispixel wird ein Stoß- oder Anziehvektor für jedes Segment berechnet. Eine Schwelle der Pixelanzahl wird aus der gesamten Anzahl der Pixel in der Umgebung berechnet (Durchschnitt). Wenn die Anzahl der Pixel innerhalb eines Segments die oben genannte Schwelle übersteigt, drückt eine Kraft die Ameise weg von diesem Segment und zwar abhängig von der Anzahl der Pixel. Jedoch, wenn die An-

zahl der Pixel im Segment unter der Schwelle liegt, wird die Ameise in die Richtung dieses Segments gezogen. Zusätzlich hat die Richtung der Segmenten relativ zum Ziel eine groše Bedeutung. Segmente, die näher an der Zielrichtung sind, stošen die Ameise stärker ab als Segmente, die weit weg von der Zielrichtung sind. So war es der Ameise möglich, Hindernisse zu vermeiden, aber immer noch die Zielrichtung im 'Kopf zu behalten'. Die Experimente mit der virtuellen Ameise wurden in drei Gruppen unterteilt. Die erste Gruppe beinhaltete Hindernisse mit einer einfachen Struktur, wie eine Wand oder eine viereckige Säule. Diese Gruppe wurde hauptsächlich dafür verwendet, um das Laufverhalten der Ameise zu testen,bei der nur ein einziges Hindernis die Trajektorie der Ameise beeinflusst. In der zweiten Gruppe wurde das Laufverhalten der Ameise im Bezug auf Hindernisse simuliert, die mit echten Ameisen getestet wurden. Beispiele sind ein Kanal, zum Ziel diagonal liegende Wände oder ein Modell der Umgebung in Alice Springs. In der dritten Gruppe wurde das Laufverhalten der Ameise im Bezug auf Hindernisse simuliert, die ein lokales Minimum darstellen sollten, wie eine sehr lange Wand oder eine Sackgasse. Mit dieser Arbeit ist es gelungen zu zeigen, dass mit Hilfe eines relativ einfachen Modells, das Laufverhalten der Ameisen zum Teil simuliert werden kann. Bei manchen Hindernisformen wurde jedoch gezeigt, dass Ameisen auch komplexere Vermeidungsmuster aufweisen können im Vergleich zu der hier präsentierten virtuellen Ameise. Bei sehr langen Hindernissen oder bei Hindernissen, die wie eine Sackgasse konstruiert sind, läuft die virtuelle Ameise in ein lokales Minimum. Dies bedeutet, dass die Ameise durch die Kräfte, die Hindernisse und das Ziel auf die Ameise ausüben, in einem begrenztem Raum gefangen wird. Bei den echten Ameisen wurde so ein Verhalten nicht beobachtet. Weitere Versuche im Bereich Hindernisvermeidung bei Ameisen müssen noch durchgeführt werden, um das Modell mit echten Ameisen besser vergleichen zu können.

Contents

2 Introduction

2.1 Overview

Well directed movement through space is an important ability in the life of most animals. While searching for food, a suitable mating partner, or a nesting ground, animals must be able to avoid obstacles blocking their desired path if they are to follow a successful course in life. For some organisms relying upon a home ground or a nest, as birds, insects or mammals, it is imperative to be able to maintain knowledge about their position in space. This ability aids such species in lightly achieving a heighly complex and sophisticated behavioral activity.

2.2 Path integration

The idea of path integration describes the ability to navigate between an origin point and a destination point in space, while constantly calculating changes in distance and movement angle relative to the origin point. With the aid of simple triangulation equations applied to the egocentric angular movement and the accumulated distance walked, ants are able to navigate back home over unknown territory not covered during their outbound journey.

As was shown in multiple researches (e.g. [6], [7], [13] and [25]) desert ants are capable of using compass information derived from the azimuthal angle of the sun in the sky (sun compass) and the patterns of polarized light (polarization compass) in order to determine their exact orientation in space relative to an internal representation of a known polarization pattern. Further, according to preliminary tests run by Professor *Wehner's* group from the Zoological Institute of the University of Zürich, ants use a very complex kind of step counter or a so called *pedometer* relying on proprioceptive cues in order to accumulate their walked distance. In one experiment all 6 legs of ants were shortened by cutting an equal bit of the ant's tarsus, which caused ants to underestimate their walking distance (*Markus Knaden*, personal note). Although several experiments have been made to find out how exactly the ant's pedometer works, scientists are still unable to give a clear answer in that matter. Wohlgemuth et. al. 2002 [32] were able to show in a series of sophisticated experiments testing the pedometry of *Cataglyphis fortis* in the vertical dimension, that ants must use more sophisticated means than merely counting the number of walked steps. In those experiments ants had to walk within an array of uphill and downhill channels and were later tested on a flat terrain. Those experiments could show that while ants are foraging and homing across uneven terrain, they project their incremental path segments onto the horizontal and perform a path integration, as if on a virtual x-y plane. That means that ants indicated distances corresponding not to the actually travelled distance but to the ground distance. Such integration tactics are of course highly efficient since *Cataglyphis* ants rely highly on path integration and might forage and home on two different paths: the outbound path could differ substantially from its three-dimensional structure compared to the inbound walking path, causing a miscalculation when only using a step counter.

A few years later, *S. Sommer* and *R. Wehner* [21] were able to show that an ant's path integrator is constantly dependent upon its compass component. In these experiments, ants were allowed to walk inside channels that were partially covered by a tiny roof. In the roofed segments, the ant's celestial compass was temporarily rendered useless as no sun information was available in those sections. *Sommer* and *Wehner* had discovered that ants used to undershoot the position of a fictive nest when walking

through a tunnel with partially roofed section. They concluded that "*...the ant's path integrator had not been able to process the distance information acquired by the odometer, because there was no simultaneous input from the compass, that is no directional information accompanying the distance information...*". Figure 2 shows a diagram of an ant's inbound and outbound walking paths.

In naval language, this concept is more known as *dead reckoning* and was used for centuries by mariners in order to determine their ship's position regardless of lacking visual orientation points. By identifying well known landmarks like a charted lighthouse or a cluster of rocks, mariners were able to compare their calculated position with their true position and when needed adjust their calculated course to fit the ship's actual position at sea. In the flat and bare world of desert ants, as in the case of *Cataglyphis fortis*, using landmarks is often not an option. These ants integrate walked distance and angular data regarding their walked path in order to obtain information about their current position relative to the point of departure [25]. Another method of orientation used by ants is utilizing pheromones secreted from an exocrine gland located in the posterior abdomen to mark a walked path as attractive. An ant on a foraging round would only lay such a pheromone track if walking back towards its nest and only when its foraging run was successful. Many ant species use such path pheromones to lay down tracks for other colony members. Those colony members can detect such path pheromones with the aid of their antennas and so follow the existing path. Since trail pheromones are volatile (undetectable by ants about 80 seconds after secretion) a constant and dense stream of ants is needed to keep such pheromone trails intact [31]. Due to that fact, desert ants that are active during the hot periods of the day are unable to effectively use pheromone trails because the heat of the desert floor makes pheromones evaporate much more rapidly than in cooler environments or daytimes [6].

A further mechanism used by several ant species is the tandem run. In this kind of social behavior, used for communication about new nesting grounds or the location of a feeding ground, two ants follow each other a very close distance, with the following ant signalizing its presence to the leading ant by touching its gaster and hind legs. If the physical contact is broken, the leading ant stops and waits with its gaster erected, producing a scent signal that helps the following ant regain its former tandem position [9]. For an illustration of ants performing a tandem run see figure 1. Based on the fact that *Cataglyphis* species are lone scavengers and avoid using the mechanisms mentioned above [23], they are forced to rely on other methods of navigation like path integration, and visual landmarks for an effective orientation in their environment. So summing it up, path integration actually provides *Cataglyphis* with a life-line connecting it to its nest. This life line is constantly being updated and is the inverse of the vector that is pointing towards the current feeding site when the ant is currently found in the feeding location[25].

2.3 Obstacle avoidance

By definition, an **obstacle** is *"Something that impedes progress or achievement"* (Miriam-Webster Dictionary) [1] or *"Something that stands in the way or that obstructs progress (lit. and fig.); a hindrance, impediment, or obstruction."* (Oxford English Dictionary)[2]. For an organism that is moving through space an obstacle would represent something that cannot be crossed or an object that would significantly slow down the organism's

[1] Can be found at http://www.m-w.com/
[2] Can be found at http://dictionary.oed.com/

Figure 1: Two ants performing a tundem run. The following ant taps with its antennas on the gaster of the leading ant, constantly signalizing its presence. The leading ant is informed about the desired goal and will lead the following ant to it. When the goal is reached, the following ant may become a tandem run leader, guiding other 'naive' ants. Photo by Stephen Pratt from the Department of Ecology and Evolutionary Biology, Princeton University. Adopted from http://www.princeton.edu/∽spratt/ants.htm

motion. Some examples for such obstacles would be a big body of water or mountains in the case of terrestrial organisms or patches of land, corrals or underwater ridges for aquatic life forms. For that reason, animals navigating in a cluttered environment patched with obstacles are forced to adjust their trajectory according to their surroundings in order to avoid obstacles on the way to their desired goal. As many other activities living animals are forced to perform, avoiding obstacles has its energy and time costs. Sometimes it is possible to pass through an obstacle but the energy or time invested in this action might be more "expensive" than simply making a detour around it. As in the case of the Australian desert ant *Melophorus bagoti*, approaching an obstacle like a tussock or a rock might mean risking its own life since such shaded places are often a preferred hiding and resting place for predators like spiders or lizards [11].
A further factor is the ability of ants to avoid danger. Moving objects and other organisms might endanger ants whether by squashing them or by hunting them (in the case of predators) so by avoiding such dangers an ant is more likely to reach its full life span.

2.4 The aim of this study

This study aims at modeling the behavior of desert ants, while encountering obstacles on the way to their nest or a feeding location. In order to achieve this goal a virtual model of **path integration** and a virtual model of **obstacle avoidance**, based on real-time vision, will be developed and the integration of both models will be tested. The work presented here is based on scientific work in the field of navigation and obstacle avoidance in desert ants and robots. Knowledge collected in a number of experiments was used to construct a three-dimensional (3D) computer simulation that tries to reconstruct the walking behavior of desert ants. This model offers some possible solutions in the form of algorithms that might be in use by desert ants for solving the task of obstacle avoidance.

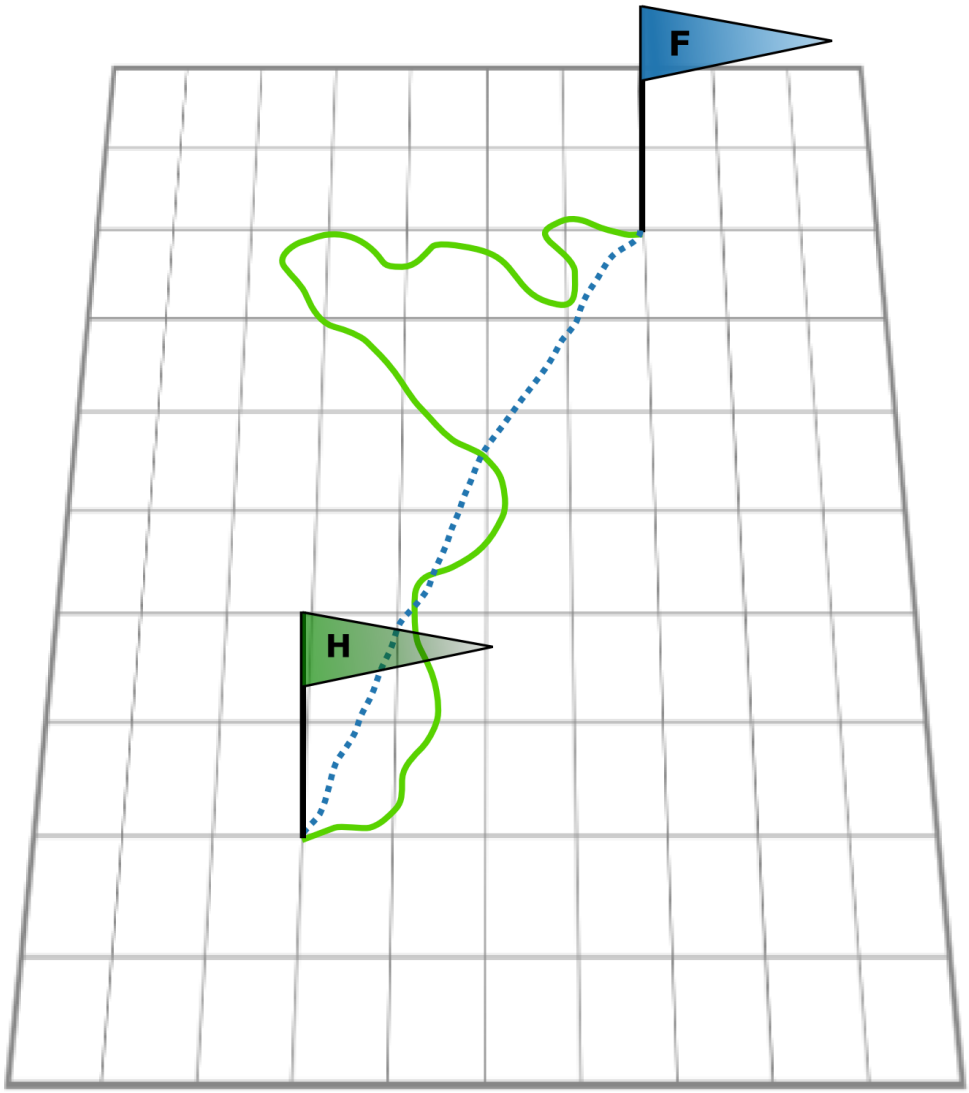


Figure 2: A diagram of an ant's foraging run. The green flag marked with 'H' (for 'Home') designate the origin point from which the ant departs on its foraging run. The blue flag marked with 'F' (for 'Feeder') designates the feeding location of the ant. The solid green line represents the outgoing path. This path is characterized by search trajectories resulting in a curved and tortuous walking path. The dashed blue line is the ingoing walking path. This path is characterized by straight trajectories since ants use the integrated home vector for returning to the nest on the shortest path possible.

3 Materials and methods

3.1 Real-world properties

3.1.1 The environment

The 3D-environment used in this thesis was reconstructed from a patch of land in the central Australian grassland (23^o43 S 133^o44 E), 17 km west of Alice Springs, which is also the natural habitat of the Australian desert ant *Melophorus bagoti* known as the Red Honey ant. This area is covered by tussocks of Buffel grass, *Cenchrus ciliaris*, which are distributed randomly over the area. The tussocks are about 10-60 cm in height and the ground is a sandy and rather flat surface of clay. Further vegetation in the surrounding area were widely dispersed trees of *Acacia estrophiolata* (Ghost gum) and *Hakea eyreana* [12]. Figure 3 shows an overview of the landscape where the experiments were run.

Figure 3: An overview image of the landscape where the testing environment near *Alice Springs* was mapped. Photo by Dr. *Ken Cheng*, adopted from: http://galliform.bhs.mq.edu.au/ ken/kcheng/Alice%20Springs/Mbagoti.html

Data from research on the desert ant species *Cataglyphis* was widely used here. This species occurs in several dry Saharan regions like in the southern parts of Tunisia, where Professor *Wehner's* research group experimented with it near the village of Maharès (34.58^o N 10.50^o E). This area, covered with flat salt pans, is sandy and is mostly devoid of vegetation or other landmarks [16]. On the fringes of the salt pans and in the habitat of several *Cataglyphis* species, some vegetation like *Chenopodiacean* shrubs of *Salicornia europaea*, *Suaeda monoica* and *Arthrocnemum perenne* can be found [29].

Figure 4: *Melophorus bagoti*. Photo by *Ajay Narendra*, adopted from: http://photos1.blogger.com/blogger/4187/1246/1600/IMG_2642.0.jpg

3.1.2 Real-world ants

Since the agent in the 3D-environment is a model based on real-world desert ants, I will discuss some of the morphological properties of those ants that were used for constructing algorithms in the simulation.

The genus *Cataglyphis* (including, among others, the species *fortis*, *bicolor*, and *albicans*) resides in the vast plains of the Tunesian desert. *Cataglyphis*, a desert scavenger belonging to the subfamily *Formicinae*, is a purely day-time active creature that forages solitarily for carcasses of other arthropods that did not succeed in retreating to a cool shelter and were killed by the heat of the desert sun. The habitat of the *Cataglyphis* species mentioned here is characterized by either low-shrub halophilic vegetation or by bare desert ground devoid of any artifacts [11]. The distance of such foraging rounds may rise up to a few hundred meters from the ant's nest [23]. With its unique body structure *Cataglyphis* is highly adapted to the scorching heat of the Sahara's desert floor. Its very long legs (compared to other species) keeps it far from the ground and so it leaves its nest when the temperature of the ground surface raises to 35-45^{o}C and is able to forage the desert floor while surface temperatures rise to above 60^{o}C. During a foraging run its body temperature may rise to above 50^{o}C, which is very close to its critical thermal maximum (53.6$\pm$0.8^{o}C), a fact that might explain their nimble and agile way of movement that is imperative for their survival [27]. A close relative of *C. fortis*, the species *C. bombycina*, is known to spend 30-75% of its foraging time on 'thermal refuges' by climbing to the tops of stalks of vegetation in its habitat ([1] and [28]). This distance from the hot desert floor raises the survival chances of the ant by enabling a somewhat 'cooler' living environment. By doing that *C. bombycina*, can lower its body temperature; for if not, it risks death of heat shock [4].

Measurements show that *Cataglyphis fortis* ants are capable of translational move-

	fortis	bicolor
Monocular visual field (ster)/(% of unit sphere)	6.63/52.8	6.40/50.9
Binocular visual field (ster)/(% of unit sphere)	1.54/12.3	1.71/13.6
Blind region (ster)/(% of unit sphere)	0.85/6.8	1.48/11.8
Rostral tip of binocular region. $\alpha(^o)/\varepsilon(^o)$	0.0/-45.0	0.0/-45.0
Caudal tip of binocular region $\alpha(^o)/\varepsilon(^o)$	180.0/5.0	180.0/25.0
Length of binocular region (^{o})	220	195
Maximal width of binocular region (^{o})/$\varepsilon(^o)$	52.0/50.0	51.2/80.0
Maximal width of blind region (^{o})/$\varepsilon(^o)$	52.0/50.0	51.2/80.0
Count of tested individuals	6	13

Table 1: Data of the visual fields of the species *Cataglyphis fortis* and *Cataglyphis bicolor*. See Figure 5 for a three-dimensional illustration of the visual field of *C. fortis* and *C. bicolor*. Adopted from [23]

ments of around 0.70 m/sec and rotational velocities of around 4000^o/sec thanks to its ability to adjust its metasoma to an upright position, a fact that helps it to cope with the centrifugal forces involved in such rapid rotational movement. It is assumed that even such quick movements are not representative of the maximal values actually achieved by ants of this genus [23]. Further, *Cataglyphis*, in contrary to many ant species, is characterized by not laying pheromone trails or recruiting other ants after finding a feeding ground. Relying on this fact, it has been shown that *Cataglyphis* uses mostly visual information in addition to path integration for orientation within its habitat. Its eyes, flat-cambered compound eyes located on the sides of the head, cover only a relatively small area of the ant's head. Each eye covers almost exactly one hemisphere of the visual globe. Both monocular visual hemispheres are tilted towards each other in a way that creates a binocular region that can reach at maximum 50^o in width and stretches rostraly-caudaly in the visual field above the ant's head [33]. A second important area in the ant's field of view is the ant's blind region stretching rostraly-caudaly and is located below the ant's head. Figure 5 illustrates the shape of the visual field of *Cataglyphis bicolor*, a close relative of *C. fortis*, and table 1 contains some detailed information about the properties of the visual field in both species.

Just as its distant relative *Cataglyphis*, the Australian ant *Melophorus bagoti*, belonging to the subfamily *Formicinae*, is one of thirteen known species of the genus *Melophorus*, the furnace ants, that is endemic to the Australian continent [2]. *M. Bagoti*, like his distant relative a thermophilic solitaire scavenger, forages for dead arthropods at extremely high temperatures. During the winter months *M. bagoti* avoids activities above ground. Among its related species it is classified as the largest and apparently best heat adapted. As was often observed, *M. bagoti* does not show any use of trail pheromones, not even when many ants are making the same foraging round from one nest to a specific single feeder. It has been observed that ants exhibit slightly different

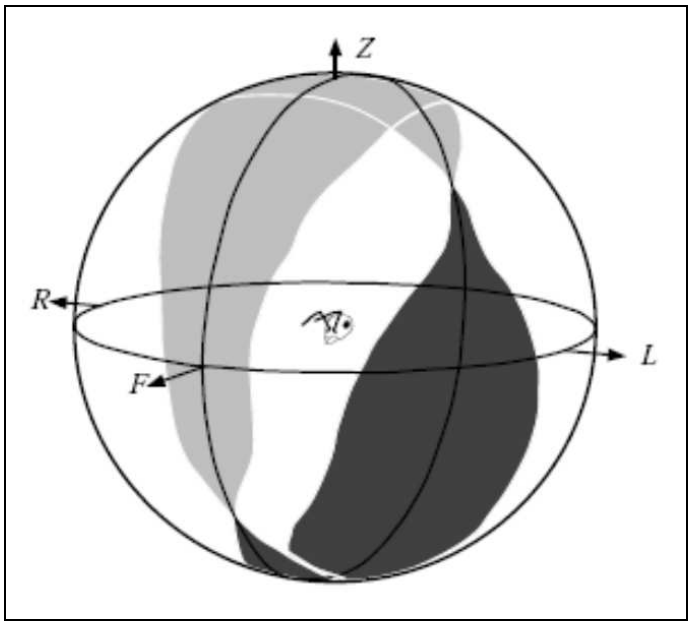


Figure 5: Properties of the compound eye's visual field of a C. bicolor worker onto the unit sphere. Light grey - area of binocular overlap; dark grey - blind region; F stands for 'frontal', L/R for left/right (lateral) and Z for zenithal. Adopted from [33]

walking paths and that the inbound and outbound foraging paths of individual ants usually differ from each other [12].

3.2 Virtual-world environment

The 3D-environment model including sky, ground surface, and the objects simulating grass tussocks was created by *Kai Basten* in cooperation with the Wehner group (the coordinates of the tussocks in the testing environment were kindly delivered in digital form by *Martin Kohler*). For creating the modeling environment the 3D-software *Creator v2.5.1* (*MultiGen-Paradigm Inc.*) was used[3]. Figure 6 shows an example of the 3D-environment. The section modeled in the 3D-environment had only tussocks (polygon outlines) in it because the mapped patch of ground did not have any trees on it. As a consequence, none of the trees found in the broad natural environment can be found in the 3D-model. The original 3D-model was later used to create derivate models for the purpose of isolating certain obstacle types in the environment and testing the reaction of the agent to only those certain model constructions. Such a working method made it easier to test and analyze the reactions of the agent to obstacles. All movements in the 3D-environment were controlled and executed by the programming language C++ using *openGL* (Open Graphics Library) libraries. The *openGL* platform is an *application programming interface* (API) for writing applications that produce, amongst others, real-time 3D computer graphics [4].

[3]For more information about *Creator* and *MultiGen-Paradigme* please see http://www.multigen-paradigm.com/products/database/creator/index.shtml

[4]For more information about *openGL* please see http://www.opengl.org or alternatively http://en.wikipedia.org/wiki/OpenGL

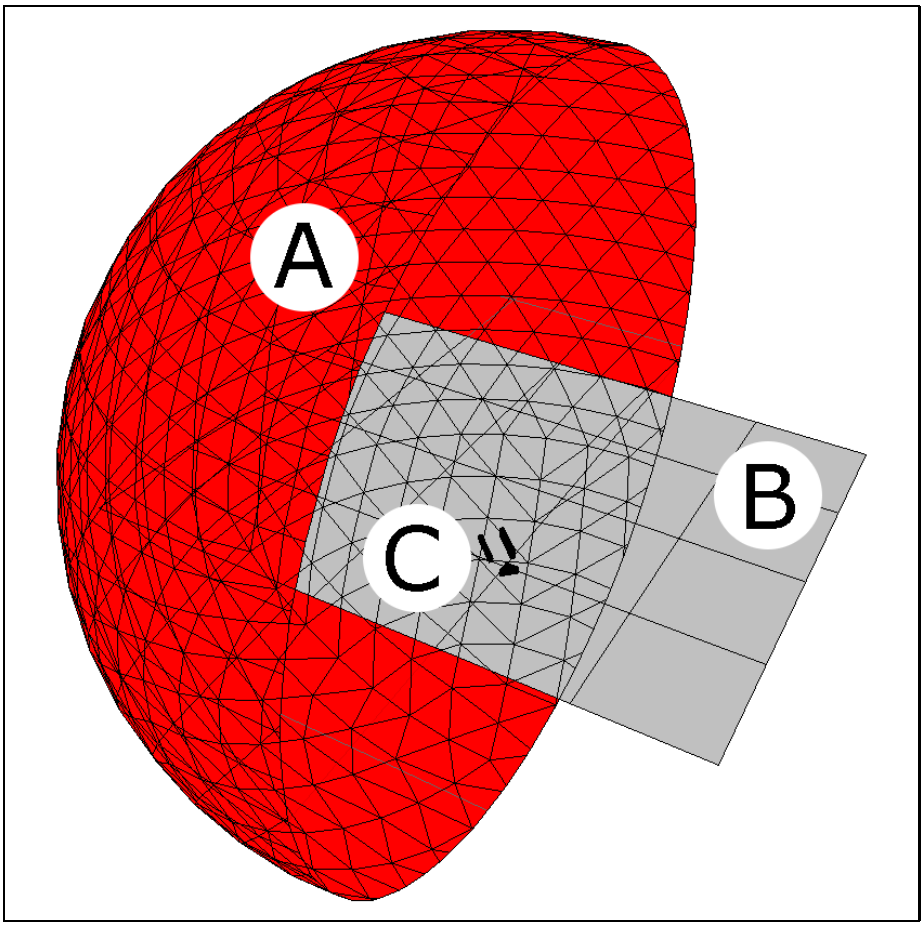


Figure 6: An illustration of the 3D-environment as used in one of the experiments. (A) The red sphere was used as the sky of the 3D-environment. A crosssection was made for demonstration purposes only; in the experiments the sky was a complete sphere. (B) The ground surface on which the agent completed its foraging rounds. (C) Obstacles as two parallel walls that were used in one of the experiments.

Basic assumptions regarding all components in the 3D-environment:

- Light in the 3D-environment was a flood of ambient light. This means that there was no defined direction from which light shone upon objects in the environment so objects in the 3D-world cast no shadow. During the whole time of using the 3D-model there were no changes made to the illumination.
- Colors of objects in the virtual-world were defined by the RGB color model. This model is an additive model in which *Red*, *Green* and *Blue* are combined in various ways to reproduce further colors. A color in the RGB model is described by a series of three numbers (here referred as RGB(x,y,x)) indicating how much of each of the red, green and blue colors, the so-called *primary colors*, is included. Each color can vary between the minimum (no color) and maximum (full intensity). If

all the colors are set to minimum intensity the result viewed is black. If all three colors are set to maximum intensity, the result is white. In the RGB model the color values are written as numbers in the range between 0 and 255. By this logic the color red, for example, is represented as RGB(255,0,0). That means that red is presented by its full intensity, green and blue have no intensity in this example [10] [22].

- In order to simplify the model only, three colors were used to represent the three different artefact types existing in the 3D-world: *sky*, *ground* and *obstacles* (i.e. tussocks). This simplification is supported by scientific work done by *Ralf Möller* in 2002. In his paper "*Insects Could Exploit UV-Green Contrast for Landmark Navigation*", he demonstrates that a color-contrast mechanism, involving the UV and green receptors of insect eyes, can deliver a robust foreground/skyline detection under varying conditions of illumination without the necessity of adjustable detection threshholds [15]. Möller built an apparatus constructed of two spectral channels: UV and green. Light falling through two interference filters (UV: 350nm, green: 500nm) was projected by two lenses onto two photodiodes. Both receptive fields of the photodiodes were almost completely overlapping (horizontal overlap $> 93\%$) at a large distance (approximately 10m), and for small objects, care was taken that the measured object was within the visual field of both sensors. 260 samples of sky patches and 368 terestrial samples were collected and were visualized in a log G and log UV diagram. As can be seen in figure 8, the two measurement groups can be almost perfectly separated from each other by a single linear threshold (slope $\omega = 0.73$), even though the intensity of both groups varies over nearly three orders of magnitude. A similar, more advanced model developed by *Kai Basten* uses more realistic color textures for artefacts in the virtual environment so the grade of realism in the model can, in case needed, be expanded later.

  By living in a dynamic environment, ants are constantly subjected to changes of illumination in their vicinity. Clouds that obscured a feeding ground may drift away or the constant change in sun angle may change the illumination of the surrounding area dramatically. Using his device, Möller was able to deliver a plausible model for distinguishing between foreground objects like plants, rocks, etc., and the sky as background over 3 illumination magnitudes. Although the above model doesn't attempt to analyze or explain the ability of insects to distinguish between foreground objects for themselves, it is possible to see, by looking at figure 8, that ground objects can be distinguished from each other even if the discrimination is not fixed for certain objects. That would mean that an ant might not recognize a certain object as the same object over different illumination magnitudes, but it can distinguish it from other ground objects. In addition to Möller's model, Chittka could show with a series of model calculations that the spectral properties of the three types of bee photo receptors are able to almost perfectly distinguish between flowers and their surrounding backgrounds like leaves or soil [5].

  Based on the two models mentioned above, it is fairly plausible to assume that real-world ants distinguish foreground from background objects and even foreground objects from themselves. By those means, the virtual ant in the simulation is also capable of distinguishing three different object types from each other.

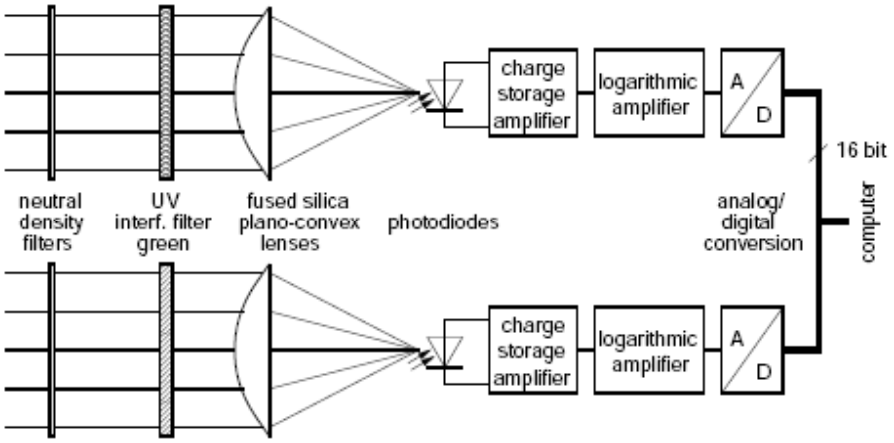


Figure 7: Schematic Illustration of the UV-green sensors used by *R. Möller*. Adopted from [15]

3.2.1 Sky

As noted earlier, the 3D-environment was constructed of three main components. One of them, the sky, was constructed of a complete sphere wrapped around the 'known' world, which were the parts of the 3D-model visible to the agent. By putting the floor of the virtual world inside this sphere, it was possible to emulate the existence of sky in the virtual environment. The color chosen for the sky was red (RGB(255,0,0)) and the size, shape and position of the sphere stayed constant during the whole development and testing period.

3.2.2 Desert floor

A flat plate covered with the color white (RGB(255,255,255)) was used as the ground on which the agent performed its foraging runs, so it was theoretically possible to fall off the edge of the world in the 3D- environment. Still, all foraging runs of the agent were made well within the premises of stable ground to avoid "accidents".

3.2.3 Nest and feeding locations

Both nest and feeding point (referred to here as *feeder*) were virtual. No actual polygons were used to materialize either of those locations. The only indication of the nest or the feeder was the software internal notation of their position in cartesian coordinates that were transformed by the path integration algorithm to polar coordinates. For more information see section 3.3.1 on page 23.

3.2.4 Obstacles

The obstacles in the 3D-environment were constructed of rectangles, cylinders or other multi-sided polygons that were positioned upright relative to the ground and upon which black color (RGB(0,0,0)) was applied. See figure 9 for an illustation of obstacles in the simulated environment. The height and width of the obstacles varied through the different experiments. All obstacles were hollow and their walls were not solid.

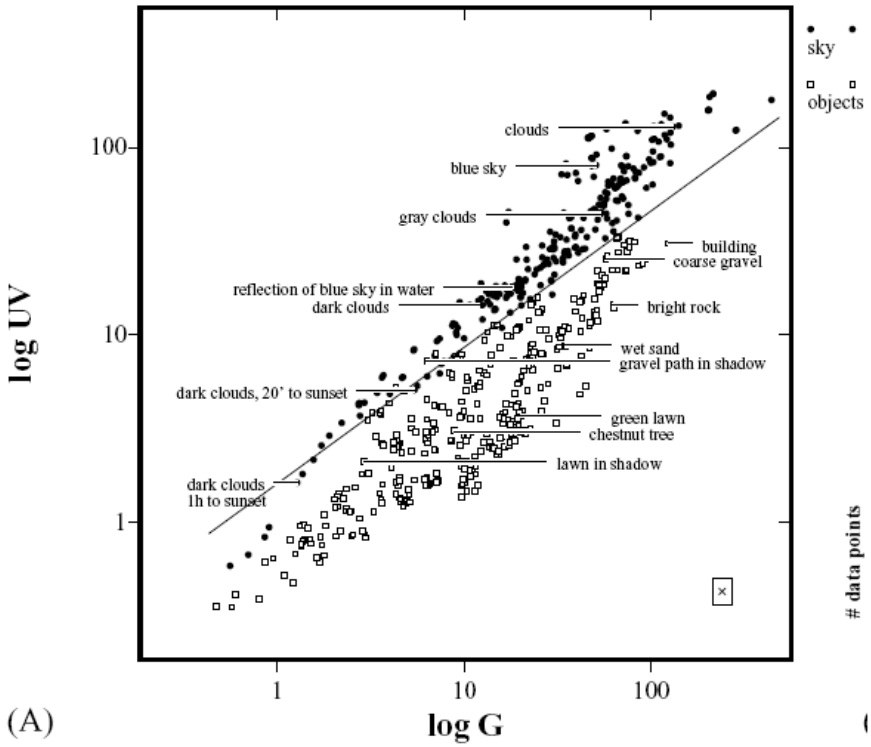


Figure 8: Samples obtained from the UV/green sensor by pointing it towards the sky and towards different objects. The labeled object samples were taken under direct sunlight, or, if labeled accordingly, in the shadow on a clear day. The object point labeled 'building' stems from a building with a stone cladding. The threshold line (slope $\omega = 0.73$) was determined so that a minimum of points is mis-classified (22 of 628). All measures were made by *R. Möller*. Adopted from [15]

That means that an agent was able to walk through an obstacle in case its avoidance algorithm guided it directly into the obstacle. For behavior of real-world ants towards obstacles please see subsection 3.1.2 on page 14. Color was applied to obstacles only on the outer surface so if an agent were to walk through and inside an obstacle it would not see the obstacle it is in, just as if the obstacle did not exist at all.

3.2.5 The agent

There were no polygons used to create a 3D-being in the virtual environment so the virtual ant, which is referred to as the 'agent' was bodyless. The agent is constructed of five cameras with an opening angle of 90^o each. All cameras originate from the same point in space at a height of 0.05 m from the ground surface and were aimed to the front, left, right, back and top of the agent. This constellation of cameras would give the agent a full 360^o field of vision if the obstacle avoidance algorithm hadn't have limited it. The agent's visual field was similar to the one in desert ants but with no binocular regions. For an illustration of the vision field of desert ants please see section 3.1.2 on page 14. The images from all five cameras were being mapped to one panorama

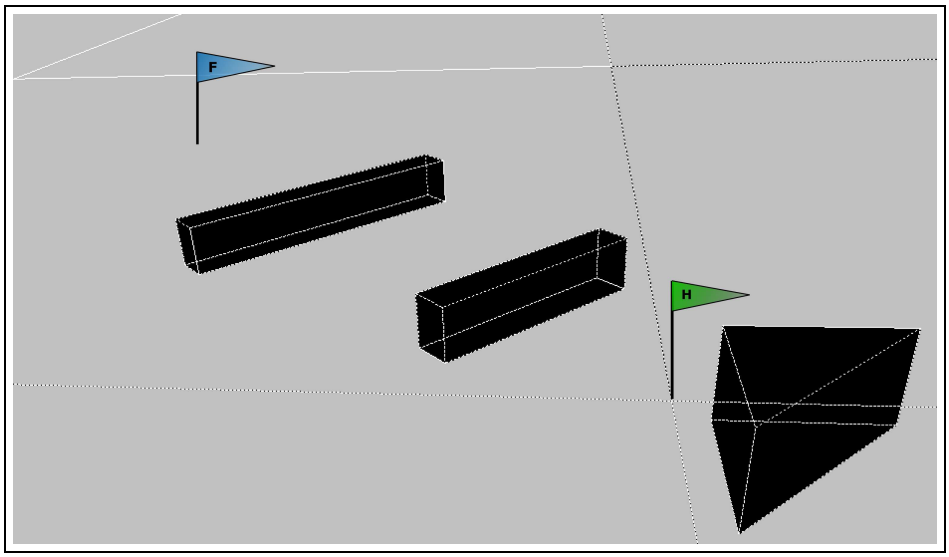


Figure 9: This illustration shows an example of obstacles as they were aligned in the 3D-environment. The two elongated walls on the left side of the figure are actual obstacles the agent had to avoid. The triangle on the right side of the figure was positioned for localization purposes of the nest's location while testing the model. For the actual experiment in which the above constellation was used see section 4.2.2 on page 39.

image that was later used for image processing in the obstacle-avoidance algorithm. The panorama created from the five cameras was not a perfect panorama though. Since the resolution of each camera image is lower on the rim of the image and since this discrepancy compared to the inner parts of each image was not adjusted, one gets a slightly distorted panorama image. The top camera image was cut into triangles and added to the top of the panorama. For an illustration of the way camera images were mapped into one panorama image please see figure 13.

There were three different ways of moving the agent around in the 3D-world. It was possible to steer it by use of the keyboard, a joystick or the more 'natural' way - it can autonomously move by itself when given a target in the form of a vector (angle, length). Variable translation velocities (step unit) of the agent, meaning step length over a period of time, could only be produced by moving the agent with the joystick, neither the movement by keyboard nor the autonomous movement were programmed to enable variable agent speeds. In other movements produced by the keyboard or with the autonomous movement model, the translation speed of the agent (agent's step length) stayed constant through the whole experimenting phase. While developing the obstacle avoidance module, joystick and keyboard steering were used, but while actually testing the agent in the obstacle environment, only the autonomous steering mode was used.

3.2.6 An internal compass

The fact that information of an internal sky compass is available to desert ants is long known [24]. Specialized ultraviolet receptors located in a small part of the compound eye, the *dorsal rim area* (DRA) of each of the ant's eyes looking at and above the horizon, construct the ant's polarization sensitive system. This system provides the ant

with compass information regarding its polar angle [25]. Based on this knowledge, the agent in the 3D-environment was also equipped with a compass (the openGL's global variable Shared->h, 'h' stands for 'heading') meaning it was constantly and continuously aware of its position relative to the global 0^o (a predefined 'north' direction). Figure 10 illustrates the polarized pattern observed by ants.

The agent used a 180^o navigational system that split equally the 3D-environment into a positive and a negative half. So for example a rotation of 90^o to the left when starting at the global 0^o would change the direction to $+90^o$. Accordingly a rotation to the right with the same value would change the direction to -90^o. Performing a turn that would change the value of Shared->h to more than $+180^o$ or less than -180^o would cause an overflow that can be represented by the following equations:

$$\varphi' = \varphi + 360^o, \qquad \forall\ \varphi < -180^o \tag{1}$$

For a negative overflow angle and for a positive overflow angle, where φ is the overflow angle and φ' is the corrected angle:

$$\varphi' = \varphi - 360^o, \qquad \forall\ \varphi > +180^o \tag{2}$$

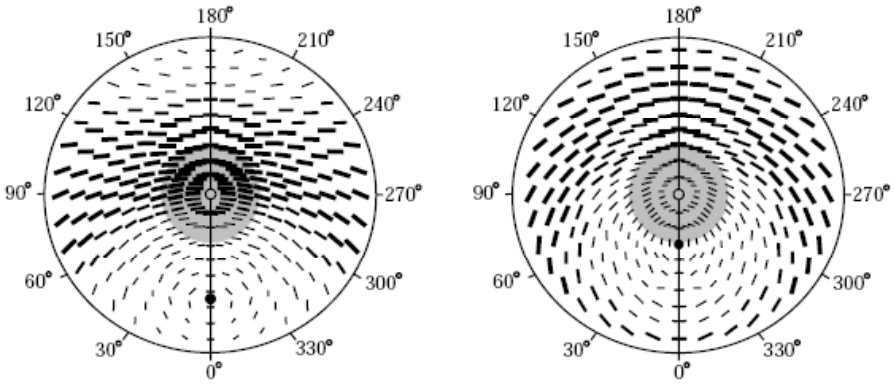


Figure 10: A two-dimensional representation of the pattern of polarization in the sky for two different elevations of the sun (black dot). On the left a sun elevation of 25^o and on the right of 60^o. The e-vector directions and the degree of polarization are indicated by the orientation and width of the black bars, respectively. For source please see [14]

3.3 Algorithms

Before delving into a more detailed visualization of the algorithms' structure implemented in this work, here is an informal definition of what is an algorithm:

> "*...an **algorithm** is any well-defined computational procedure that takes some value, or set of values, as **input** and produces some value, or set of values, as **output**. An algorithm is thus a sequence of computational steps that transform the input into the output[8]...*"

An algorithm could also be considered as a defined recipe for solving a certain problem. In that sense each of the algorithms implemented here were designed for solving a certain task i.e. the task of maintaining **path integration** of the walking agent and the task of **avoiding obstacles** the agent comes across while navigating.

3.3.1 The path integration algorithm

As mentioned in section 2.2 on page 9, path integration is a way of using simple triangulation for maintaining knowledge about the egocentric position relative to the departure point, while navigating through space. The aim of this section is to demonstrate how such triangulation was used in the path integration algorithm designed for this work. An examples for such triangulation will follow later.
The agent was equipped with two path integration vectors that were calculated and stored separately from each other. The algorithm was iterative and each output calculated by the algorithm served as the input of the next step. The first vector was the home vector that constantly maintained navigational information as distance (meter) and as direction (arc degree) relative to the location of the agent's nest and the second one was the feeding vector calculating the location of the feeder. Both actions happened only whenever the agent was in motion and both vectors were updated by the same algorithm implemented as a single computer function. This function got alternately inputs for the home vector and for the feeder vector. So as professor Wehner observed in experiments done with real-world ants "*...the vector pointing home was always the inverse of the one pointing towards the feeding site...it was merely the sign of the vector that was reversed once the ant has arrived at the nest or the feeder...*"[25].

The home vector is initially set to zero length and zero degrees in direction when the agent is in the nest, which was the starting point for every test run. When leaving the nest the home vector was newly calculated for each step iteration whenever the agent performed a movement. So the navigational information contained in the home vector constantly pointed to the nest, which means that if the foraging run was successful and the agent should have walked towards the nest, all it had to do was turn to the direction of the home vector angle and walk the distance stored in it.
The feeder vector is initially set to contain the distance and the angle corresponding to the feeding site as if the agent already visited the feeding site at least one time. Since a food search algorithm was not implemented in the model, there was no other way of 'teaching' the agent were food could be found. Thus outbound foraging paths always led directly to the feeder unless there were some obstacles in the way of the agent.
There was of course the question of motivation. Should the agent walk home? Is it currently on a feeding run? Those motivation questions were handled by a boolean valued variable also known as *flag* that was used as a status indicator set in the beginning of each test run. When the agent leaves the nest the flag is set to 'feeder mode' meaning its motivation is to gather food so it followed the navigational information stored in the feeder vector. When the feeder position is reached the flag is switched over and the agent changed its motivation: now it was interested in reaching home and 'unloading' the food previously collected. When reaching home the state of the flag is switched over and again the agent followed the feeder vector's navigational information and so on and so forth. Independent of the current motivation of the agent, both home and feeder vectors are constantly updated, while the agent is performing rotation and translation movements.

Figure 11 illustrates a detailed triangulation sequence of an agent's walking path, while encountering a tussock represented by a green deformed object blocking the direct

path to the feeder located between the points C and A2. This obstacle forced the agent to change its initial direction to the target (in that case the feeder) as well as to update both of his path integration vectors. For simplification reasons I will try to explain the whole triangulation process by breaking it into more basic steps.

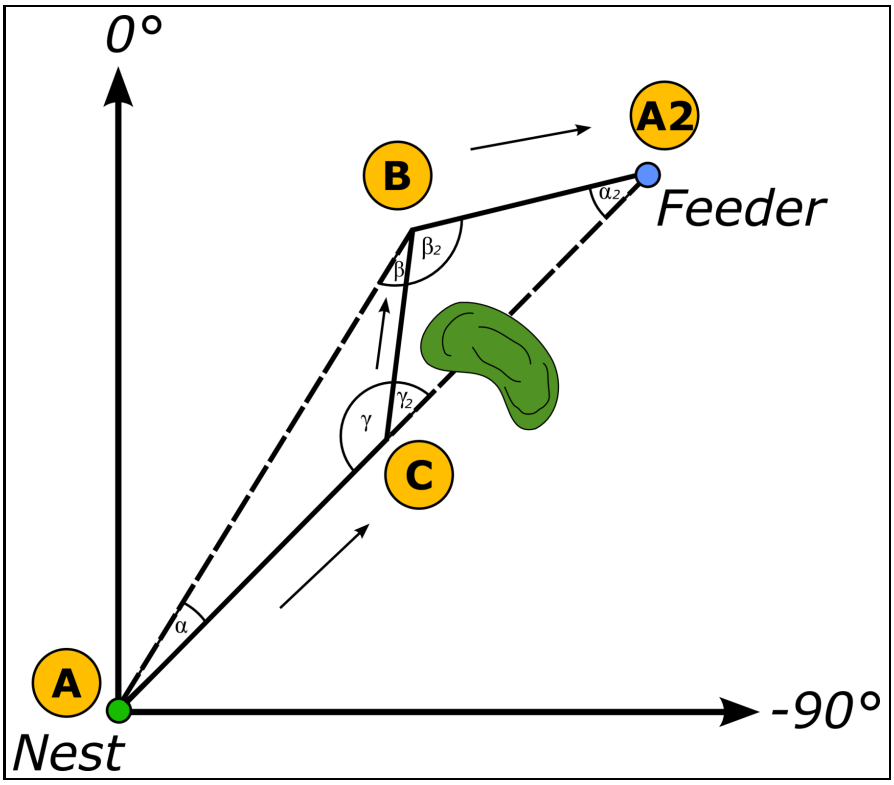


Figure 11: A detailed illustration of the way the agent performed triangulation sequences for updating the state of both home and feeder vectors. The green deformed shape between point C and A2 is an obstacle shaped as a tussock blocking the agent's original feeder path ($\overline{AA2}$). The black arrows indicate the agent's walked path. The black solid lines are the actual walked path and the dashed lines are non-walked paths that correspond to the original feeder vector and the updated home vector, respectively. The triangle leg $\overline{BA2}$ represents the newly calculated feeder vector and the leg $\overline{BA}$ represents the newly calculated home vector.

After the first step was taken by the agent (A→C) there was still no need to apply triangulation. Because no rotational movement was combined with the translation, the only change to the home vector, here designated as $\vec{\mathbf{v}}_H$, is the extention of the vector's length. So by adding the current step length $\overline{AC}$ to the initial length of the home vector (it is 0 in size since this is the first step taken) the home vector is now updated according to the following equation:

$$\| \vec{\mathbf{v}}_H \| = 0 + \overline{AC} \tag{3}$$

On the contrary, the feeder vector, here designated as $\vec{\mathbf{v}}_F$, is shortened. This action

is demonstrated by the following equation:

$$\| \vec{\mathbf{v}}_F \| = \overline{AA_2} - \overline{AC} \tag{4}$$

After rotating into the direction of point B and taking the second step (C→B) things become a bit more complicated. Now the agent is standing in point B meaning translation and rotation were performed consecutively. The new parameters needed for calculating $\vec{\mathbf{v}}_H$ are now the length $\overline{BA}$ and the absolute turning angle β for returning home. For the calculation of $\overline{BA}$ the leg lengths of $\overline{AC}$, $\overline{CB}$ and the size of γ are needed. The current unit step is known to the algorithms as well as the last length of the home vector $\vec{\mathbf{v}}_H$. The size of γ can be extracted by subtracting the current heading of the agent from the last heading of the agent (before taking the last step):

$$\gamma = \varphi - \varphi_{heading} \tag{5}$$

where φ is the last turned angle and $\varphi_{heading}$ is the global heading angle both known to the agent (*openGL's* intern variable Shared->h). Angles are not shown here.
For the calculation the *cosine* law is then used:

$$\| \vec{\mathbf{v}}_H \| = \sqrt{\overline{AC}^2 + \overline{CB}^2 - 2\overline{AC}\,\overline{CB}cos\gamma} \tag{6}$$

For calculating the angle β, which stands for the new direction of the home vector, the following equation is used:

$$\beta = arccos\frac{\overline{CB}^2 + \overline{AC}^2 - \overline{BA}^2}{2\overline{CB}\,\overline{BA}} \tag{7}$$

The updating process of $\vec{\mathbf{v}}_F$ in step two as well as of both vectors, after step three was taken (B→A2), is not described here explicitly since they follow the same rules seen in equations 5 through 7 over and over again. Figure 12 demonstrates a flow chart of the algorithm's structure in more detail.

Some words must be spoken about floating point errors. In cases where the triangle calculated in the path integration algorithm is very acute, i.e. if $\overline{CB}$ is small relative to $\overline{BA}$ and $\overline{AC}$ or if β is very small compared to 1, round-off errors of the simulation were encountered. Such errors occurred when the agent was walking a steady trajectory meaning the ground was clear of obstacles and the walking path was therefor a straight line to the target. In such cases the calculated triangle was very narrow and produced trajectories that sometimes missed the target. Section 6.4.5 below discusses a possible workaround.

3.3.2 The obstacle avoidance algorithm

The following section deals with the core algorithm of the presented thesis and explains the inner functionality and the logic behind the obstacle avoidance algorithm. For a flowchart schema illustrating the inner functionality of the algorithm please see figure 16 on page 34.

The obstacle avoidance algorithm is based on a real-time analysis of the viewed surroundings after every movement of the agent in the 3D-environment. Its functionality was divided into two components. The first one was to calculate rejecting forces pushing the agent away from obstacles and the second one was used to calculate pulling forces

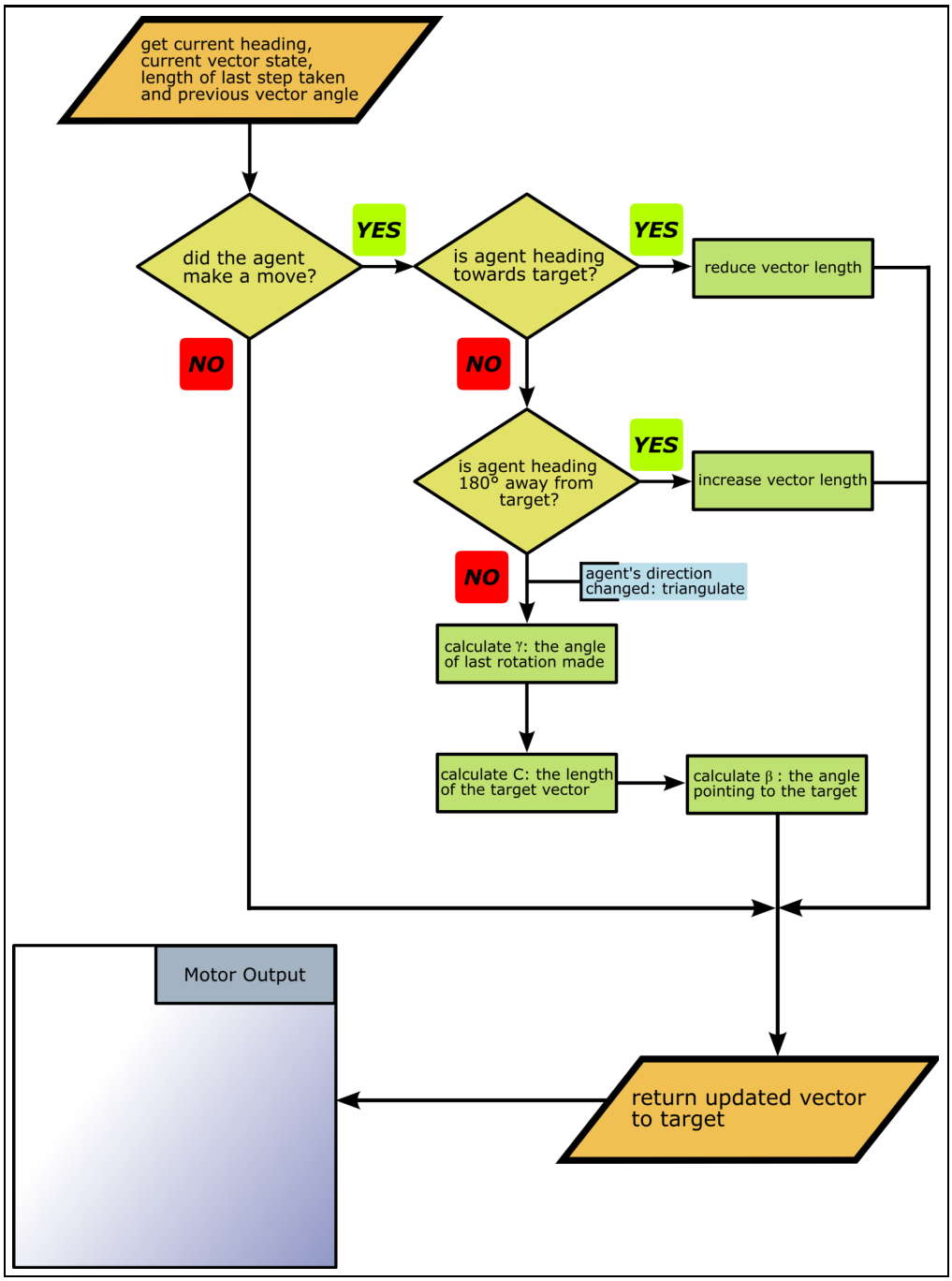


Figure 12: A flowchart illustrating the path integration algorithm. The algorithm receives from the motor output module the current agent's heading, the current state of the vector, the length of the last step taken and the previous vector angle. It outputs an updated vector containing the length and direction of the target. The above illustrated algorithm was used for updating both home and feeder vectors.

that pulled the agent into regions of low obstacle density.

As mentioned in section 3.2.5 on page 20 the agent has an initial 360^o panoramic view of the environment but this sensory input was not fully used by the algorithm. The reason for that has a simple mathematical explanation and can be demonstrated by the following example: since the panorama was 256 pixels in width and was divided by the algorithm into 19 segments (see later in this section), the calculation of unused pixel in the panorama is:

$$256\ pixels\ modulo\ 19\ segments = 9\ pixels \tag{8}$$

That is because the division of the total number of pixel in the image through the number of segments doesn't produce a natural number. Since pixels can only be equal to a natural number ($\mathbb{N}_0$) a remainder is left. As seen in the example, 9 pixels are not taken into account in the process of obstacle avoidance and are discarded. Since 2 pixels are equivalent to one ommatidium in the agent (horizon), 4 ommatidia are 'lost' in the visual field of the agent. In total, the blind region stretches over 12.65^o from the starting value of 360^o. It can be seen as the uttermost right segment in figure 13B. For a comperison with the visual field of *Cataglyphis* see table 1.
The basic assumption concerning the functionality of the algorithm is that ants are able to discriminate the skyline from ground objects and also some ground objects from themselves (see section 3.2 page 18). For that reason the algorithm suggested here concentrates solely on the detection of obstacles in its panoramic field of view and ignores all other visual queues. It does this by iterating over a complete graphic image and scanning it for certain pixel properties. In this case the algorithm was searching for black pixels RGB(x<10, y<10, z<10), which was the color of obstacles in the 3D-environment, counting them and storing them in a container (array) for later use. The panorama itself was initially sliced into 9 segments and later, in the more advanced algorithm, into 19 segments. Each segment was circa 40^o in width in the 9 segment-panorama and circa 18^o in the 19 segment-panorama. The number of segments was chosen rather arbitrarily but had to have a center (uneven number of segments). This was designed to give an equal number of segments on each side of the vision field's center. See the discussion chapter in section 6.1.1 for more detailed information about choosing the number of segments for the algorithm. A panorama image taken in the 3D-environments in one of the experiments and an illustration of how the panorama image is built out of the five cameras looking into the environment can be seen in figure 13. Figure 13A shows a panorama with 19 segments that are visualized by black vertical bordering lines added to the image after the algorithm's analysis and for testing purposes only.

The black objects, which can be seen in figure 13 are simulated tussocks and the yellow dot on the right side of the panorama designates the direction of the target used for visualization reasons only. All panorama images were identical in size and were 256 pixel wide and 96 pixel high. The initial idea of deviding the panorama into a small number of segments was adopted from a robotic model developed and tested by A. Ohaya et. al. [17]. In this model *Ohya* and colleagues developed a mobile robot that processed in real-time an image taken from a single video camera mounted on the robot. With the aid of this image the robot was able to self-localize and detect obstacles standing in its way. The self-localization mechanism was based on simple odomentry and on the image of the environment previously taken by the robot (since robot odometry is notorious for its accuracy). By comparing both images the robot was able to determine its actual location and to correct it accordingly. By using an edge detection filter the robot was

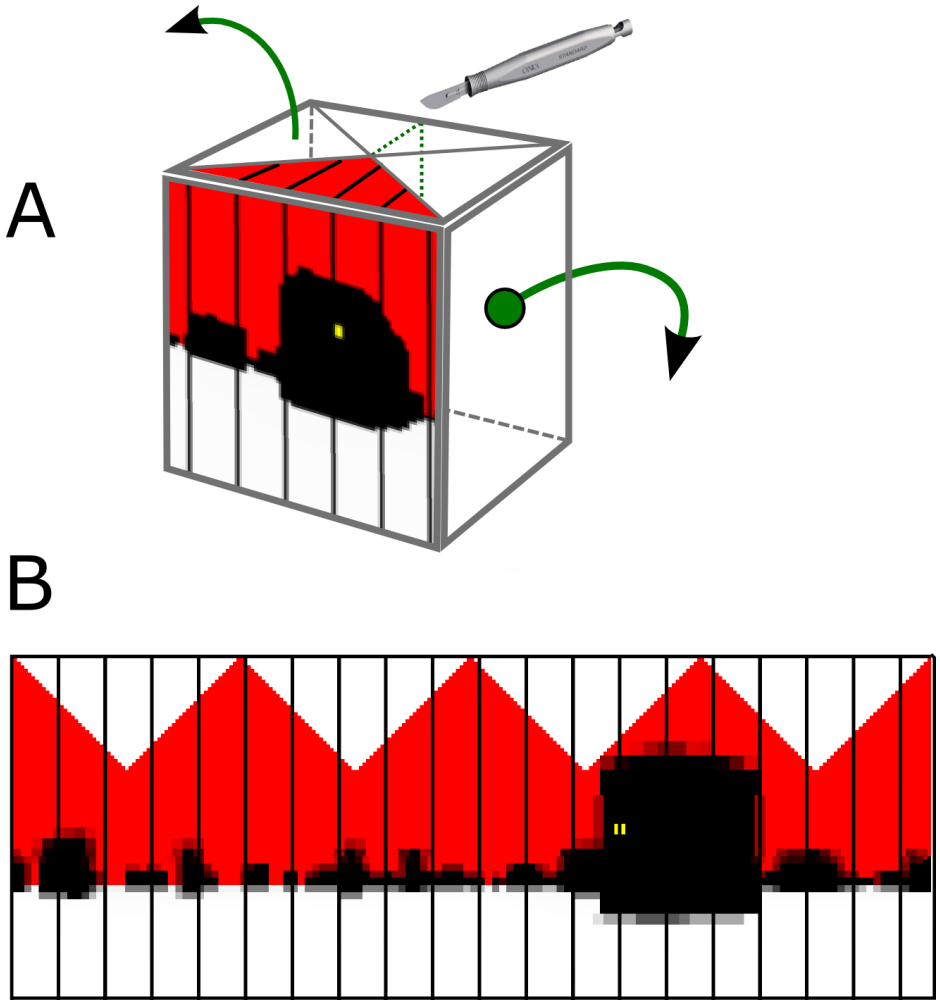


Figure 13: (A) The array of cameras in the 3D-environment. The green dotted lines on the back of the cube show where the cube is sliced to create the panorama seen in figure 13B. The cube image was reconstructed from a work by *Kai Basten*. The uttermost right segment is the blind region in the agent's visual field. The image of the scalpel is adopted from the company *Canica Design Inc.* and can be originally viewed on the following link: http://www.canica.com/chess-scalpel.asp. (B) The 19-segment panorama. A snapshot of the 3D-environment taken in one of the agent's foraging runs. The obstacles seen here are of the cluttered environment as it was mapped in the vicinity of *Alice Springs*, Australia. The red regions (above the horizon) is the sky and the white regions are the ground of the 3D-environment. The yellow dot seen on the obstacle to the right hand side, points at the direction of the target. The black vertical lines did not occur in the original image and were added later by the algorithm for visualization purposes only. Segments seen in the center look to the front of the agent. Segments closer to the panorama edges look behind the agent.

able to detect objects and decide whether they block the path lying before it. This was done by dividing the image into segments and counting the detected edge pixels in each of those segments. If segments in the path of the robot contained a number of edge pixels that was above a predefined threshold then those segments were declared as unsafe and the robot initiated an avoidance maneuver around them. Because the robot had previous images of the environment, it was able to differentiate between edges of known (non-obstacle) objects and 'new' unrecognized objects blocking its way.

In the model presented here the agent used three components in order to calculate an avoidance angle that would enable it to avoid obstacles standing in its way to the target:

- The number of black pixels in each segment
- The direction of each segment
- The direction of the target

Computing the segment's rejecting force

The avoidance angle, as produced by the rejector, is computed according to equation 9:

$$\vec{\mathbf{v}}^*_{r,i} = s_i \mathbf{e}(-\varphi_i) w(\varphi_i - \varphi_T) + \mu \mathbf{e}(\varphi_T) \tag{9}$$

$\vec{\mathbf{v}}^*_{r,i}$	The resulting avoidance vector (rejector)
i	Segment position
s_i	Sensor input. The length of each segment's vector (pixels count per segment)
φ_i	The angular value to which each segment is pointing (relative local zero)
φ_T	Target's bearing
w	Weight given to each segment (relative to the target segment)
μ	Weight of target vector's length

Table 2: A list of the variables occurring in equation 9

In general, computing a single rejection force induced by the obstacles in the visual vicinity of the agent was preformed by summing up the rejecting vectors created by each segment. Segment vectors were calculated by counting the obstacle pixels in each of the segments (vector length) and computing it with the angle of that segment (vector heading). After counting the black pixels for each segment, representing the length of the vector (s_i) in each of the segments (referred to as *sensor reading*) a single movement vector for the next agent's step was calculated out of the pixels count and the predefined angle of each segment φ_i. φ_i was determined by deviding 360^o with the number of segments in the panorama and assigning the result to each segment successively. The angular value of each segment was always the center of the segment and so in the case of the algorithm using 19 segments, the middle segment was assigned the angular value of 0^o, the next segment to its right was -18.94^o and the one on its right side was assigned the angular value of -37.89^o and so on. Segments to the left side of the middle segment were assigned the corresponding positive values. The calculated vectors were called rejecting vectors or *rejectors* since they were forces functioning as repellents pushing the agent away from the obstacles. This effect was achieved by inversing the direction of the unit vector calculated in each of the segments ($\mathbf{e}(-\varphi_i)$).

A further factor was the weight ($w(\varphi_i - \varphi_T)$) given to each segment. Since segments facing the target have a different significance than segments opposing the target, there was a need to add weight to the resulting segment vectors in order to get the appropriate impact from each segment's vector. Because the segments turning in the direction of the target usually point to the walking direction of the agent, there was a need that those segments reject the agent stronger than segments opposing the target in case an obstacle was detected. The segments on the rear, opposite to the target, were on the other hand somewhat irrelevant for the agent since they usually represent obstacles that were already avoided, meaning their given rejecting weight was minor. Therefore a weighting was introduced as a curve that was dynamic relative to the location of the target for each walked step. In tests with 19 segments this curve (Gauss, $\mu = 0$ $\sigma^2 = 1.0$ or $\mu = 0$ $\sigma^2 = 0.8$) was sampled along 9 positive and 9 negative positions on the x-axis ($-4.5 \leq x \leq 4.5$) including a further position, the peak of the curve: $x = 0$. In tests run with the 9 segmented algorithm the curve was sampled on 4 positive and 4 negative positions ($-4.0 \leq x \leq 4.0$) and on the zero position: $x = 0$. This curve was used for factoring each of the segment vectors by calculating the peak of the curve with the target segment first and only then all the other points of the curve with the rest of the segments relatively to the target segment. See figure 14 for an illustration of the Gauss function used for weighting the segment vectors.

After the vectors from all segments were calculated, they were added to each other as well as to the target vector. It should be mentioned that the target vector $\mathbf{e}(\varphi_T)$ was initially a unit vector, which length could be adjusted by the factor μ. The μ factor was manually determined before each test run. The angle of $\mathbf{e}(\varphi_T)$ was the target bearing as stored in the corresponding home or feeder vector. The distance to goal played no role concerning the forces the goal had on attracting the agent. There is no evidence of ants having a higher motivation when near a target. The outcome of this calculation was a single vector resolving the best direction to walk to, considering all visually detected obstacles in the environment and the direction of the target (equation 11). From this vector only the angle was used and the vector length was discarded since the agent in the model had a constant length of step unit. It can also be assumed that using the length of the avoidance vector, the vector calculated by the obstacle avoidance algorithm, would create movements that may not respect nonholonomic constrains, thus generating avoidance movements that a real-world ant could never produce.

Computing the segment's attracting force

Later, when it was discovered that some obstacles could not be avoided by this kind of algorithm, a slight modification of the initial algorithm was implemented. The difference between the new and the old algorithm was that in the old algorithm empty segments containing no black pixels (i.e. obstacles) did not have any force what so ever, meaning that those segments did not have any effect on the walking trajectory of the agent. Not so in the newly implemented algorithm. The new algorithm includes a threshold (condition: $\forall s_i > \overline{s}$) for the sensor reading represented by the pixels count in each segment ($\forall s_i$). This threshold ($\overline{s}$) was determined dynamically by continuously calculating the mean value of all pixels counted over all segments for each walked step. This gave the agent an indication of how dense the environment was, which determined how appealing 'empty' segments were classified. When moving through an environment with a spacious distribution of obstacles, the agent would consider a segment appealing only if the intensity of the visual sensor's input (black pixel count) is very low. On the other hand, when the agent is moving through an environment, which its obstacle dis-

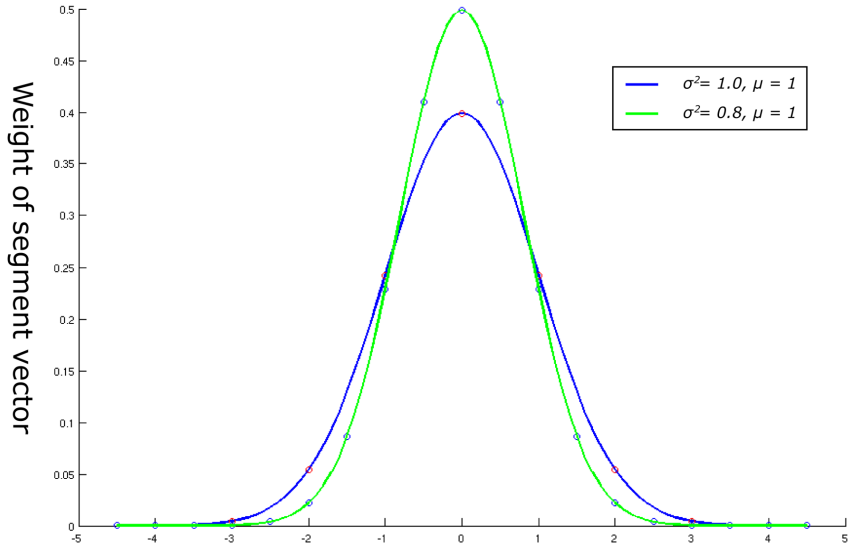


Figure 14: The Gauss curves that were used to extract weight values for the segments. The ∘ sign on each curve indicates the points that were sampled along the curve. In this illustration the blue (lower) cureve was sampled 19 times. The green (higher) curve was sampled 9 times. When used with the 19-segments algorithm, both cureves were sampled 19 times.

tribution is dense it would consider segments with a relative high amount of black pixels to be appealing. By that sense, when the environment becomes crowded with obstacles, the agent is less fastidious in classifying a segment as attractive. If the threshold of a certain segment was exceeded, that would mean that either there is a rather big obstacle in this segment or that the obstacle is very close by and that a rejecting vector should be calculated for this segment. On the other hand if the count of pixels in a certain segment was below the threshold value then an attracting vector or a so called *attractor* was calculated. The attracting vector is computed according to equation 10:

$$\vec{\mathbf{v}}^*_{a,i} = s_T^2 e^{-\frac{s_i}{\overline{s}}} \mathbf{e}(\varphi_i) \tag{10}$$

$\vec{\mathbf{v}}^*_{a,i}$	The resulting avoidance vector (attractor)
i	Segment position
s_T	Sensor input at the target segment (pixel count)
s_i	Sensor input. The length of each segment's vector (pixels count per segment)
φ_i	The angular value to which each segment is pointing (relative local zero)
$\overline{s}$	Mean sensor input

Table 3: A list of the variables occurring in equation 10

The direction of the avoidance angle is then determined by the summation of all

attractors and rejectors and by extracting the angle from the received vector as seen in equation 11:

$$\vec{\mathbf{v}}_i = \begin{cases} \vec{\mathbf{v}}^*_{r,i} & s_i \geq \overline{s} \\ \vec{\mathbf{v}}^*_{a,i} & s_i < \overline{s} \end{cases}$$

$$\vec{\mathbf{v}}^* = \sum_{i=1}^{n} \vec{\mathbf{v}}_i$$

$$\mathbf{v} = \frac{\vec{\mathbf{v}}^*}{\| \vec{\mathbf{v}}^* \|} \cdot steplength \tag{11}$$

$\vec{\mathbf{v}}_i$	Component of segment i
i	Segment position
$\vec{\mathbf{v}}^*$	Sum of all rejectors and attractors
$\mathbf{v}$	The bearing of the avoidance angle

Table 4: A list of the variables occurring in equation 11

Such a vector would pull the agent into the direction of an empty segment. The output of the attractor is dependent on the number of pixels in the target segment and also on how low is the pixels count in the segment currently calculated. The force of the attractor grow to the power of 2 with an increasing number of pixels in the target segment (${s_T}^2$) and grow with a decreasing number of pixels in the own segment (starting at the threshold value). This factor ($e^{-\frac{s_i}{\overline{s}}}$) is the inverse of the exponential function to the power of pixel count in that segment divided by the mean value of all pixels in the panorama. Figure 15 demonstrates schematically the forces influencing the agent's (seen in the center of the pentagon) walked trajectory out of a bird's eye view.

3.3.3 The simulation as a whole

The computational model described here is constructed of several components. The *motor output module* or MOM, as it is later referred to, is in charge of moving the agent around while simultaneously getting fed information from the *vision module* or VM, which is in charge of obstacle avoidance and from the *path integration module* or PIM, which is in charge of updating home and feeder positions. An illustration of those modules can be seen in figure 17.

The MOM can also be regarded as the central unit where all sensor inputs are being transformed into movement in the 3D-world. The whole communication between the three components is continuous and goes always in both directions. That means that from the view point of the MOM information is flowing in the direction of the PIM but information is also flowing back from the PIM into the MOM. Accordingly, information is flowing from the MOM to the VM and from the VM back into the MOM. For the task of integrating path information the PIM requires from the MOM the current agent's heading, the last state of the current vector and the length of the last step taken by the agent. In exchange the PIM delivers back to the MOM the updated direction of the current target (be it the nest or the feeder). On the other hand the VM requires from the MOM only one angular value. This angular value is the delta value between the current heading of the agent and the bearing of the target. The visual module, like the PIM, delivers only one angular value back to the MOM, which tells the MOM how it

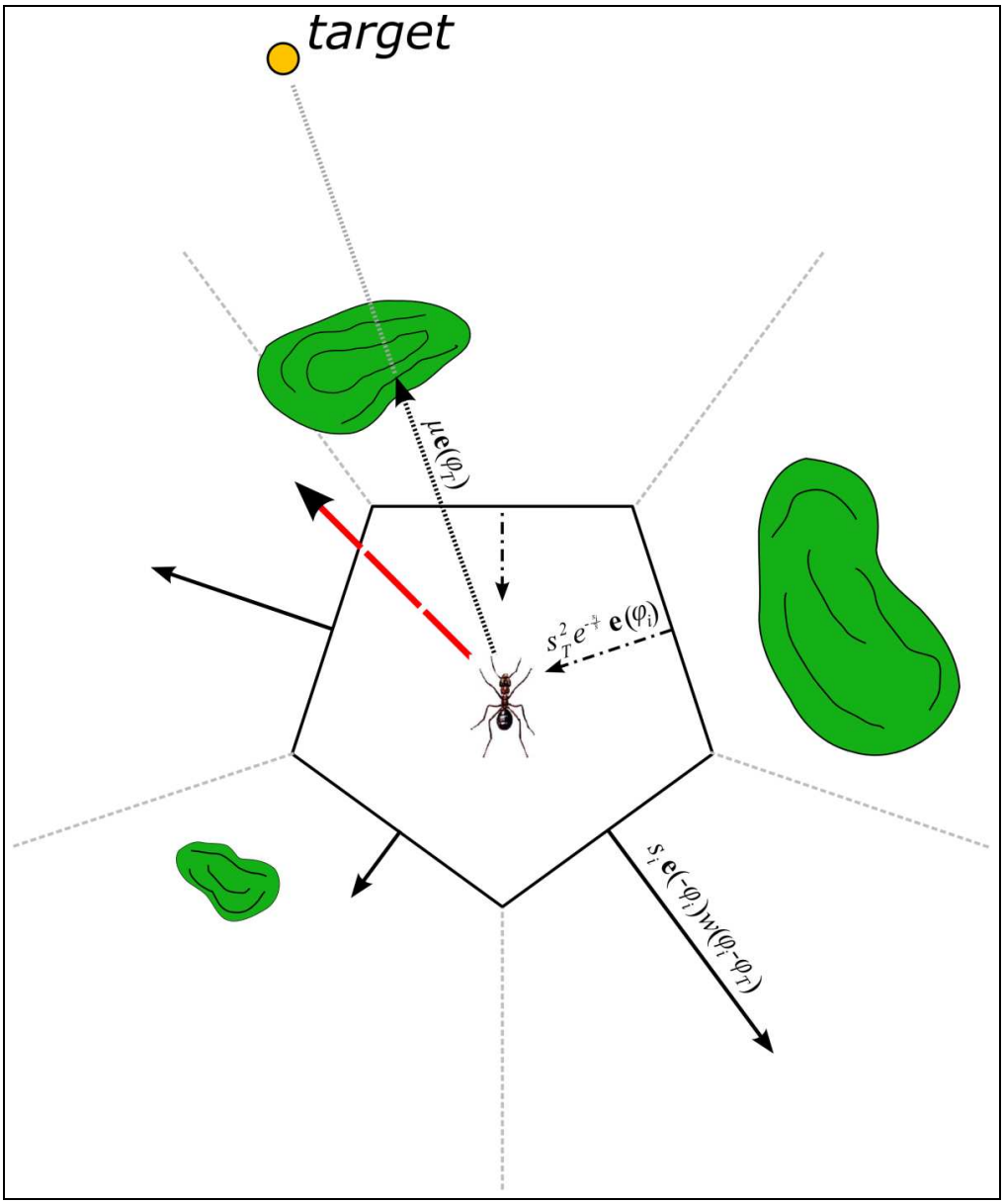


Figure 15: An illustration demonstrating the functionality of the obstacle avoidance algorithm. The segment structure seen here as a pentagon was simplified for demonstration purposes only. The original model was constructed of polygons of 9 or 19 faces. The dashed gray lines are the extention of each segment so any obstacles inside the boundaries of those lines are assigned to that segment. Rejectors can be seen as dotted-dashed arrows pointing inwards. Attractors are solid arrows ponting outwards. The deformed shapes around the pentagon are tussocks influencing the walking behavior of the agent. The finely dotted arrow pointing to the target is the target vector and the line extending it is meant to emphasis the direction of the target. The thick red dashed arrow extending from the ant is the vector calculated from combining all other vectors in the scheme. All measures in the figure were arbitrarily chosen.

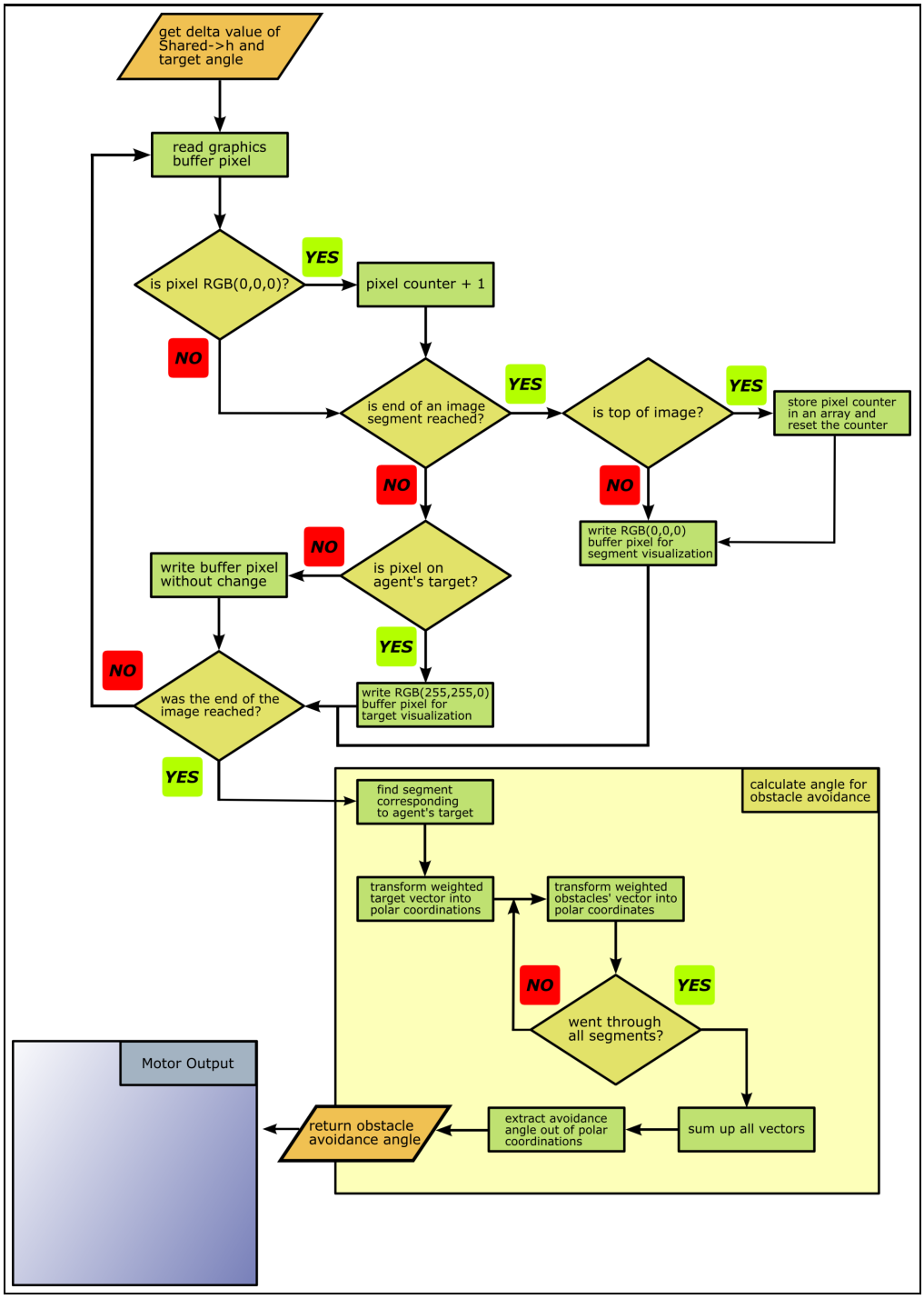


Figure 16: A flowchart illustrating the obstacle avoidance algorithm. The module calculating the avoidance angle receives from the MOM the delta value between the heading of the agent and the bearing of the target. The calculated avoidance angle is then returned to the MOM that uses it for its next move.

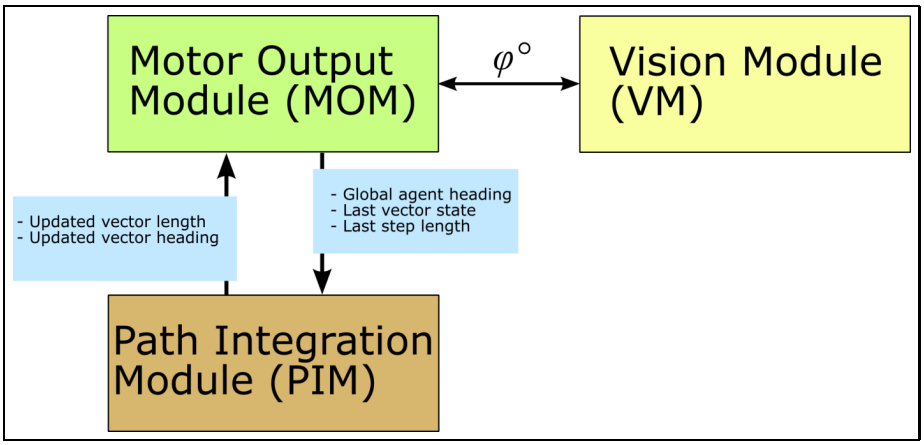


Figure 17: This illustration elucidates how the three movement modules were integrated in the virtual ant model.

should change the course so the agent best avoids obstacles in its visual vicinity.
The question asked is of course how all this comes together to create an agent that is aware of its position relatively to the nest and feeder and that moves in a way that will not drive it against any obstacles. This is best explained as a series of logical steps executed each time the agent moves:

1. Since the agent is already informed about the location of the target when the simulation starts the first action it takes is to make a step in that direction.
2. The next action taken is to update the home and feeder vectors according to the rotational and translational changes caused by the agent's first step.
3. The last action would be to look at the world and calculate a 'wise' walking direction for the next step the agent is about to take. In this step, the desired target bearing and the obstacles in the visual vicinity of the agent are taken into account. The outcome determines the direction of the agent's next step.

When the last action is carried out, the chain of events starts all over again: make a move, update knowledge about location and finally look at the world and make a wise choice about the direction of the next step and so on and so forth.

3.4 Methods of data analysis

Data analysis was performed using MATLAB® v.7.1.0.138 Service Pack 3. The x and y coordinates of the agent's movement as well as turned angles and walked distances of the agent were automatically stored by the 3D-environment. The stored data was then processed by a MATLAB® script for visualization of the walking trajectory of the agent depending on the obstacles presented in the 3D-environment. The following calculating steps were performed for each visualization plot:

1. Read raw movement data.
2. Calculate walking trajectory according to x and y coordinates. This data contained the actual walking trajectory of the agent.

3. Calculate walking trajectory according to triangulation data stored in the home and feeder vector. This was the estimated walking trajectory as calculated by the path integration module.

4. Plot the obstacles of the walked environment.

5. Plot the actual and estimated walking trajectories of the agent.

6. Store plot images for later analysis.

As mentioned above, the plots present two separated walking trajectories. The first is the walking trajectory that was registered as a cartesian coordinate system. The second was the self triangulated walking path as computed by the path integration algorithm. This path was exclusively used in directing the agent to its currently pursued target. The data about the cartesian coordinates trajectory was only used as a reference to the triangulated walking data.

4 Experiments

The walking behavior of the agent was tested in seven different environment constellations that will be presented in the following section. The environments were divided into three groups, *basic* environments, *model validation* environments and *deadlock* environments, according to their unique arrangement and shape of the obstacles tested. If not mentioned otherwise all walking trajectories of the agent originated at the coordinates (0, 0) marked in the 3D-illustrations as 'H' (home) and ended in the coordinates (-2.12, +2.12) marked as 'F' (feeder). Such trajectories simulated a direct walking azimuth of $+45^o$ (to the left of the global 0^o) and a walking distance of 3m equivalent to real-world distance. A distance unit in the environment is equivalent to one meter in the real-world.

4.1 The basic environment

Two environment constellations were constructed for testing the reaction of the agent to obstacles with a basic structure.

4.1.1 A symmetrically positioned obstacle

Constructed out of one single obstacle, this environment was often used during the development process of the obstacle avoidance algorithm. The obstacle was centered according to the direction of the feeder. That means that the left side and right side of the obstacle were equally 'divided' by an imaginary line streached from the home to the feeder so when the agent approached it, the obstacle covered a similar area on both sides of the visual sensor's center. See figure 18 for an illustration of the obstacle.

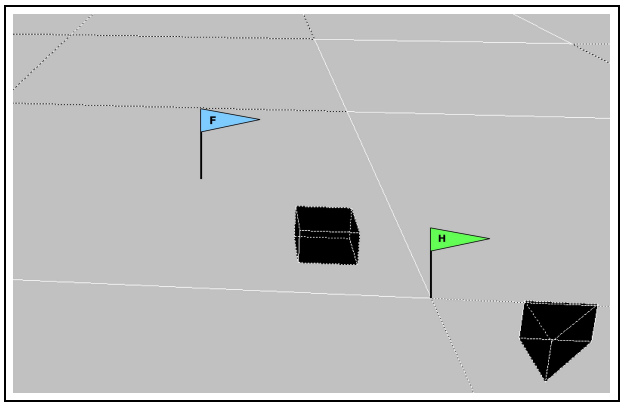


Figure 18: This image illustrates the first obstacle used in the environment. This obstacle was often used while the obstacle avoidance algorithm was in development. The green flag ('H') was the starting point of the foraging run, the blue flag ('F') was the end point of such a run. The black cube between both flags was the obstacle the agent had to avoid. The triangle object next to the green flag was used by the experimenter for information about the agnet's current heading.

4.1.2 A non-symmetrically positioned obstacle

This environment had one obstacle as in the above described environment. This time however, the obstacle was stretched significantly wider than the obstacle in the former environment. In addition the stretched obstacle was positioned such that it was offset to one direction (agent's right hand side). Due to that fact, as the agent approached it, the obstacle did not cover the agent's field of view uniformly. In other words the obstacle size was different on each side of the visual sensor's center. See figure 19 for an illustration.

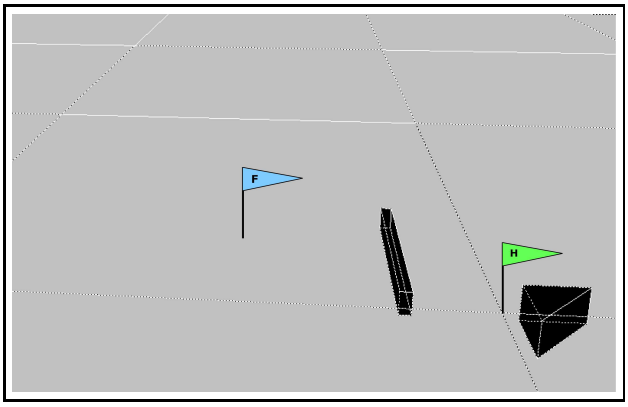


Figure 19: A single obstacle positioned not symmetrically regarding the position of the feeder.

4.2 Model validation based on behavioral data

Three environment constellations are based on behavioral data collected in field experiments with desert ants. The experiments were adopted from work done by Professor *Wehner* and colleagues.

4.2.1 The channel

Tests with a channel, in which ants of the species *C. fortis* had to walk through on their way back to the nest after a successful foraging round, were performed at the experiment station of the University of Zürich next to *Maharès*, Tunisia [11]. The channel, consisting of two walls 0.2-0.4m high and 1.5m apart, was 4m long and oriented in the north-south direction, i.e. parallel and symmetrical to the ant's homebound paths. The experiments were meant to test the centering response of desert ants to uneven heights of the channel walls. What *Wehner* et. al. observed was a centering response in ants that walked in a channel with walls of even height. On the other hand, when ants walked through the channel when the walls were of uneven heights, they exhibited a walking trajectory that tried to compensate the difference in angular elevation of both walls. That means that the ants attempted to balance the height of the image seen by the left and by the right eyes. Therefore ants tended to walk closer to the lower wall, so both walls would occupy the same sensor elevation on both eyes. For the experimental part described here, the channel walls were covered with a uniformly black color on the inner side of the channel. A schema of the channel used by *Wehner* and colleagues can be viewed in figure 20.

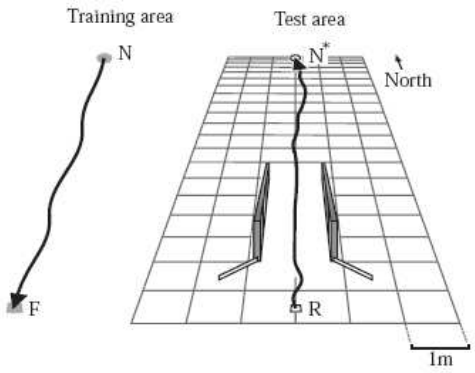


Figure 20: An illustration of the experimental arrangements: ants were trained to walk from the nest (N) to a feeding location (F) (left trail). When the route was learned the ants were transformed to the testing area (right trail), which was parallel to the training area. In the testing area the ants then had to walk from the location of the fictitious feeder (R = release) back to a fictitious nest (N*), while passing through the channel. Data by *Heusser* and *Wehner* 2002 [11].

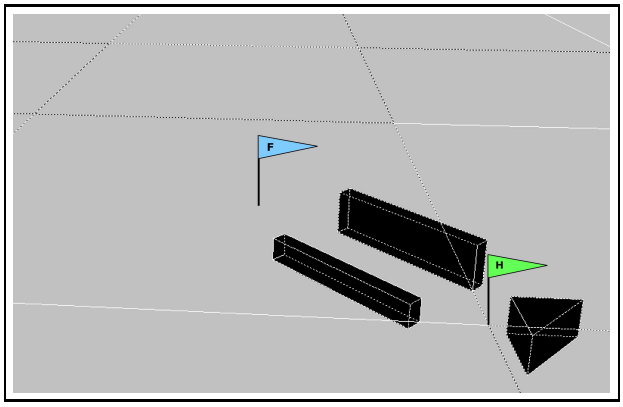


Figure 21: An illustration of the channel as it was constructed in the 3D-environment. Notice the difference in wall heights comparing both sides of the channel.

The environment described above was reconstructed in the 3D-world in three different versions (see table 5), which one of them (left wall: 0.2m, Right wall 0.4m) is illustrated in figure 21. The walls were positioned parallel to an imagineary line streached between the the starting point and end point at $+45^o$ (to the left side of the global zero). The walls in all experiments were 1.38m long and 0.67m apart.

4.2.2 The forced-detour paradigm

In the forced-detour constellation, two narrow aluminum bars, 7m and 10m in length, were used as parallel obstacles. The bars were positioned at 3m distance from each other and had a triangular profile that allowed ants to climb at one end (pointing to the nest) but not on the other end. The elevated end was perpendicular to the ground and

	Left Wall[m]	**Right Wall**[m]
Construction I	0.2	0.2
Construction II	0.2	0.4
Construction III	0.2	0.6

Table 5: Parameter of wall height used for the channel walls. The chanell was reconstructed according to the experiment run by *Heusser* and *Wehner*.

was additionally covered with Teflon® foil to prevent ants from climbing it. Ants could walk from the nest to a feeder in a straight line but were forced to make a detour while walking back to the nest [25]. Figure 22 demonstrates the experiment's constructions.

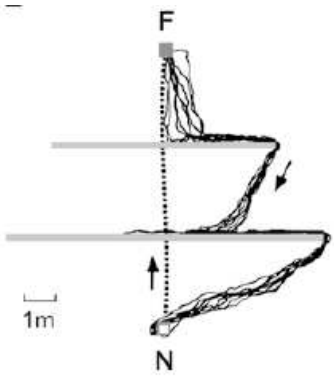


Figure 22: The forced-detour paradigm. Ants were able to walk directly to a feeder (dotted line) but had to make a detour around aluminum bars while walking back to the nest (multiple black paths). Shown are nine successive runs of one ant. 'N' represents the location of the nest, 'F' the location of the feeder. Adopted from [25]

The forced-detour constellation was reconstructed in the 3D-environment and can be viewed in figure 23. The walls, 0.83m and 1.23m in length, were 0.2m high and the gap between them was 1.07m. The obstacles had a rectangle profile since the walking trajectory of the agent was only tested in one direction. Unfortunately *Wehner* et. al. delivered no exact information regarding this experiment so it is not exactly known how high were the aluminum bars used as obstacles. Notice that in the experiment run in the 3D-environment the agent walks from the nest to the feeder but the order of the obstacles is switched compared to the original experiment. The reason is that all experiments started at the nest but still the agent had to meet the narrow bar first, as performed in the original experiment.

4.2.3 The cluttered environment

The cluttered environment is a patch of land mapped by *Kohler* and colleagues from the *Wehner's* group at *Alice Springs*, Australia [12]. For a more detailed description of this environment see section 3.1 on page 13. Figure 24 illustrates a section from the cluttered environment, taken from an experiment by *Kohler* 2005. Five successive outbound trajectories of one single ant in the real-world cluttered environment are shown.

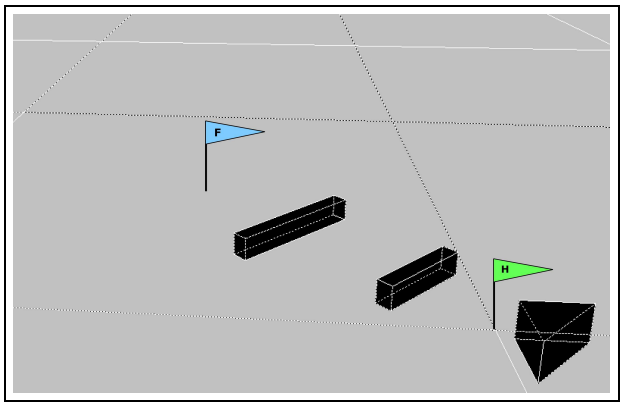


Figure 23: An illustration of the forced-detour obstacle as it was constructed in the 3D-environment. Notice the difference in wall length.

	Nest I (0.0, 0.0)	**Nest II** (+0.3, +0.3)
Feeder I (-5.0, +6.5)	(0.0, 0.0)→(-5.0, +6.5)	(+0.3, +0.3)→(-5.0, +6.5)

Table 6: For testing the walking trajectory of the agent, starting and ending points of the agent's path were changed. The above table describes the different locations of start and end points of the agent's foraging rounds used in the cluttered 3D-environment. Coordinates are set as follows: (x,y)

The exact same cluttered map was used for testing the walking behavior of the agent. Two different sets of starting (nest) points and one ending point (feeder) were used in the agent's foraging runs. Those points are described in table 6. For an illustration of the cluttered environment as it was used in the simulation see figure 25.

4.3 Deadlock environments

Two environment constellations were constructed for testing the reaction of the agent to possible deadlock situations in which the agent was unable to reach its target.

4.3.1 The infinite obstacle

This environment was constructed out of an "infinitely" long wall that stretched across the whole ground surface. The actual length of the wall was 22.7m and it streached across 4-5 segments on each side of the local 0^o meaning circa 75^o-94^o of the agent's panorama. Figure 26 illustrates the infinite wall.

4.3.2 The U-shape obstacle

The U-shape environment was created as three walls in the shape of the letter 'U'. It is illustrated in figure 27. Each wall of the 'U' construction was 0.69m long and 0.2m high.

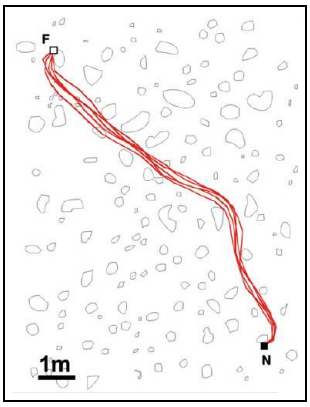


Figure 24: A section of the cluttered environment as mapped in *Alice Springs*, Australia. The drawn trajectories show five successive outbound runs of one single ant. The fine lined chunks seen in the picture are the outlines of tussocks manually mapped. Adopted from [12].

4.4 Noise generator

Calculations run by computers are deterministic. For that reason, when testing the agent in a certain environment with the same parameters over and over again the agent would always deliver the exact same walking trajectories. On the contrary, ants of the species *C. fortis* exhibit small, but consistent navigational errors that for them stay unnoticable [16]. In order to simulate the effect navigational errors have on the walking trajectory of the agent, a noise generator (NG) was programmed by *Kai Basten* and integrated into the model. The NG had influence on the motor output module (MOM) (see section 3.3.3) in such a way that normal distributed integer ($\mathbb{Z}$) values (-18, 18) were added to

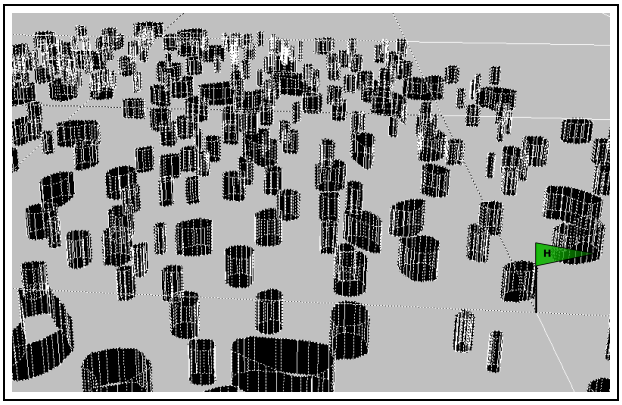


Figure 25: A patch take out of the cluttered environment as it was reconstructed in the 3D-model. One can see that the tussocks were hollow and how polygons were used to construct their round silhouette. The green flag with an 'H' represents one location of home tested (lower right side). The feeder is not marked on this image since tests were made with feeders located in different areas of the environment.

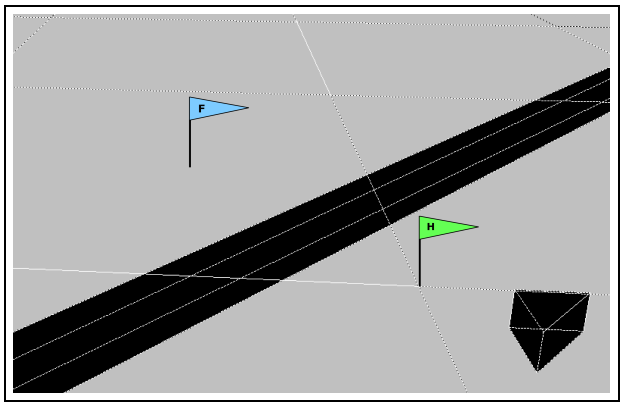


Figure 26: An obstacle as an infinite wall stretching across the whole environment's ground. The green flag with an 'H' represents home and the blue flag with an 'F' represents the feeder. The triangle object on the right hand side of the image was used for supporting the orientation capability of the experimenter.

the output of the vision module (VM) (see section 3.3.3) in charge of producing obstale avoidance angles. The noise was added to the avoidance angle independently of its size. The crucial component in his process of adding angular errors to the walking path of the agent was to inhibit the triangulation regularly performed by the path inegration algorithm since otherwise the agent would have been informed about the changes in cource.

The distribution was computed out of the probability of the ants performing a turn of a certain angle (see figure 28B) and the correlation between the angular magnitude of that turn and the error angle associated with it (see figure 28A). The outcome was

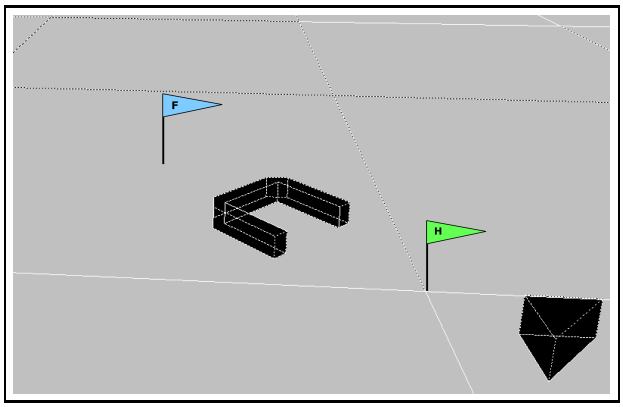


Figure 27: An obstacle constructed as a U-shape. This experimental constellation might present a non avoidable obstacle for the agent. The green flag with an 'H' represents home and the blue flag with an 'F' represents the feeder. The triangle object on the right hand side of the image was used for supporting the orientation capability of the experimenter.

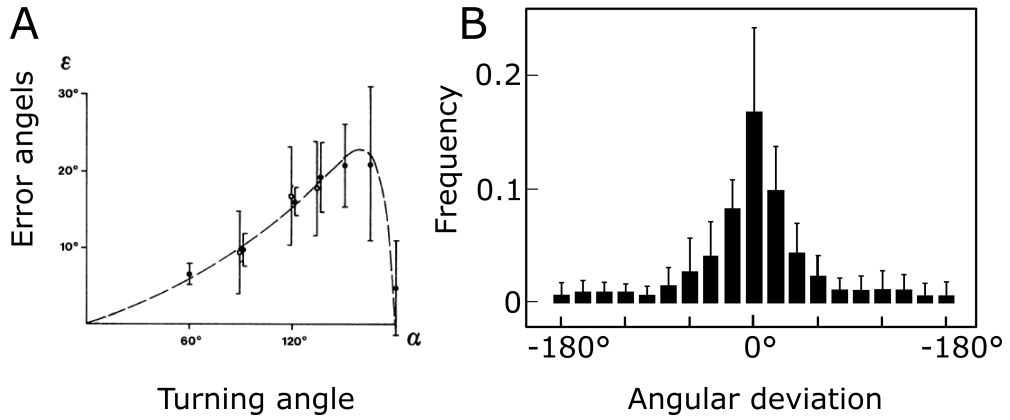


Figure 28: An illustration of the graphs used to calculate the distribution of errors the NG was using. (A) Error angles (ε) plotted for different ant turning angles (α). The ants were allowed to use either their sun compass ($\circ$) or their polarization compass ($\bullet$) system. Confidence limits are given for $P = 0.99$. The data include 1412 experiments performed with 310 ants. Adopted from [26]. (B) Freuqency distribution of turn angles in *C. fortis* as recorded in the foraging paths of 19 ants. Adopted from [30].

a Gaussian curve showing with which frequencies ants make which angular errors. An illustration of the curve used in the NG can be seen in figure 29. Section 5.4 shows the results achieved when the NG was used.

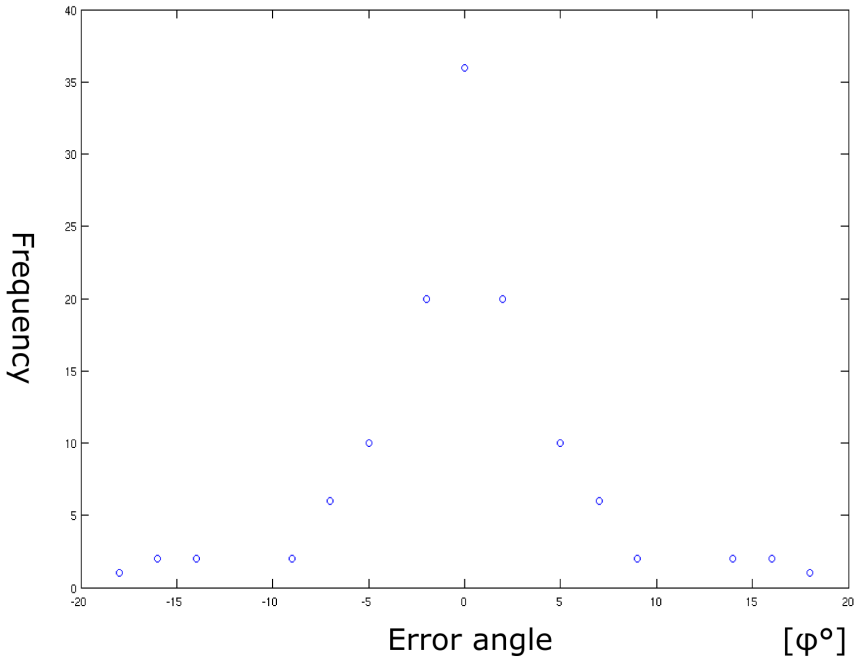


Figure 29: The noise generator. Values seen here were used in the noise generator to add errors to the agent's motor output modul.

5 Results

The agent was capable of avoiding all obstacles presented in section 4 except for the deadlock environments described in section 4.3. All trajectory plots presented here are bird's eye view two-dimensional maps of the agent's walked trajectories. The dashed gray line in each figure shown in this section is the azimuth to the target of $+45^o$ and 3m (in all the environments besides the cluttered environment). Measurement units are given in meter and the dimensions of the environments are equivalent to real-world measures. If not mentioned otherwise, all experiments were conducted using the target vector weight $\mu = 0.04$ and for the rejector the standard Gauss curve $\sigma^2 = 1.0$ was used. Table 7 gives an overview about the parameters that were used for each experiment run in the 3D environment.

Fig.	Target vector's weight	Segments' weight	Segment count
30	$\mu = 0.04$	$\sigma^2 = 1.0$	19
32	$\mu = 0.02$	$\sigma^2 = 1.0$	19
33	$\mu = 0.04$	$\sigma^2 = 0.8$	19
34	$\mu = 0.04$	$\sigma^2 = 1.0$	9
35	$\mu = 0.04$	$\sigma^2 = 1.0$	19
36	$\mu = 0.04$	$\sigma^2 = 1.0$	19
37	$\mu = 0.04$	$\sigma^2 = 1.0$	19
38	$\mu = 0.02$	$\sigma^2 = 1.0$	19
39	$\mu = 0.04$	$\sigma^2 = 1.0$	19
40	$\mu = 0.04$	$\sigma^2 = 1.0$	19
41	$\mu = 0.04$	$\sigma^2 = 1.0$	19
43	$\mu = 0.04$	$\sigma^2 = 1.0$	19
44	$\mu = 0.04$	$\sigma^2 = 1.0$	19

Table 7: A description of the different agent's parameters used in each of the experiments. The three paremeters that could be manually manipulated were the weight of the target vector, the weight given to each segment in calculating the rejecting force of objects and the number of segments that divided the visual field.

5.1 The basic environment

The agent was first tested in environments that had a simple structure. Such experiments were useful for analysing the behavior of the agent when confronted with one isolated, simply built obstacle. Both obstacles presented below were avoided by the agent regardless of which avoidance algorithm was used: 9-segmens or 19-segments (see section 3.3.2 on page 27), rejectors only (see section 3.3.2) or rejectors and attractors (section 3.3.2) combined.

5.1.1 A symmetrically positioned obstacle

The symmetrically positioned obstacle, seen in figure 30 as a square between the nest and the target, was the first obstacle the agent was tested on. The obstacle was symmetrically positioned relatively to the direct path leading to the target so when approaching it, the agent detected a similar obstacle size on each side of the middle segment. As can

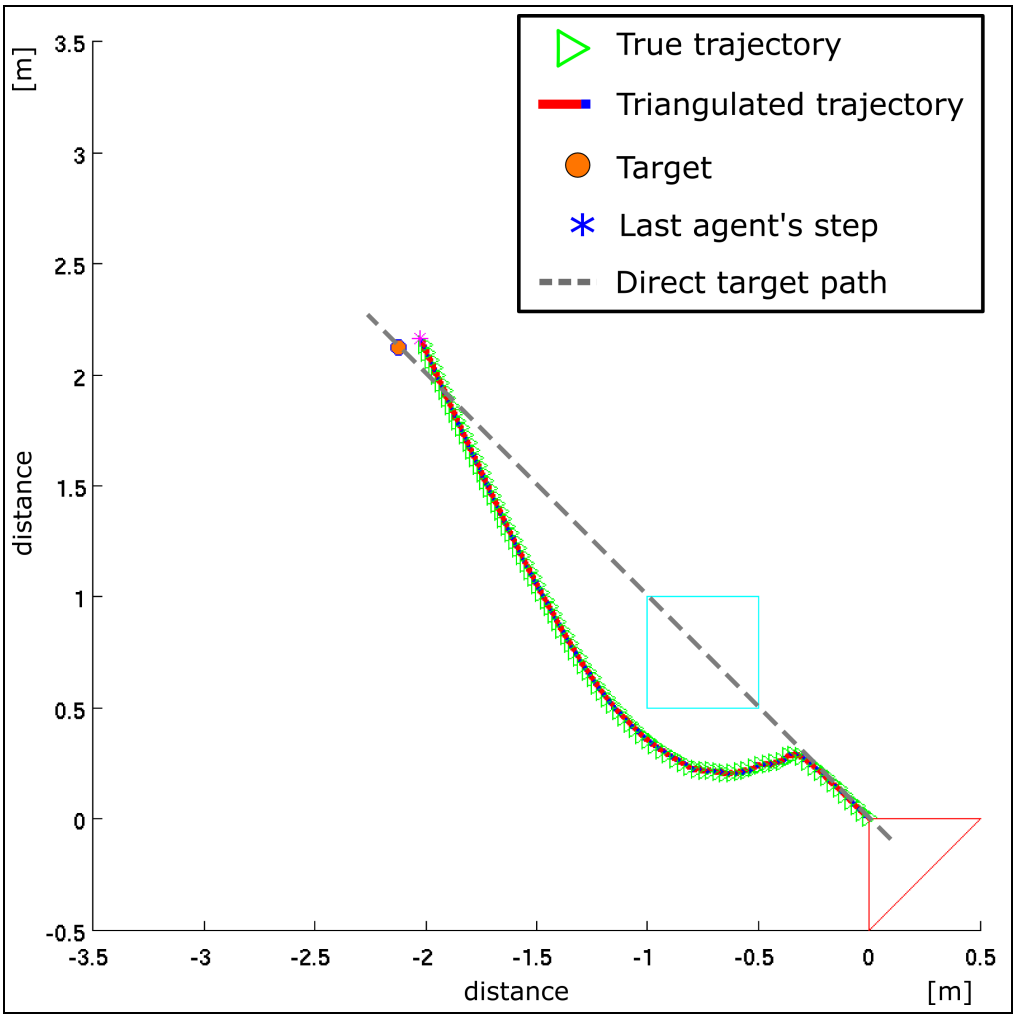


Figure 30: A single obstacle positioned symmetrically relative to the direct path to the target. The agent first reacted to the obstacle at a distance of 0.28m. It then performed a 130^o turn to the left, running an arc around the obstacle and passing its bottom left corner in a distance of 0.08m. The deviation between the last step of the agent and the target was caused by rounding errors when using computer float values. See section 3.3.1 for a more detailed information.

be seen from the agent's trajectory the agent succeeded in avoiding the single symmetrically positioned object.

Figure 31 illustrates the change in home and feeder lengths and the agent's heading compared to the heading of the home and feeder vectors. Notice the change in global heading and vector heading of both the home vector and feeder vector when the agent was engaged in avoiding the obstacle.

When the target vector weight component μ was reduced from $\mu = 0.04$ to $\mu = 0.02$ the agent maintained a bigger distance from the obstacle compared to the case in the previous experiment shown in figure 30.

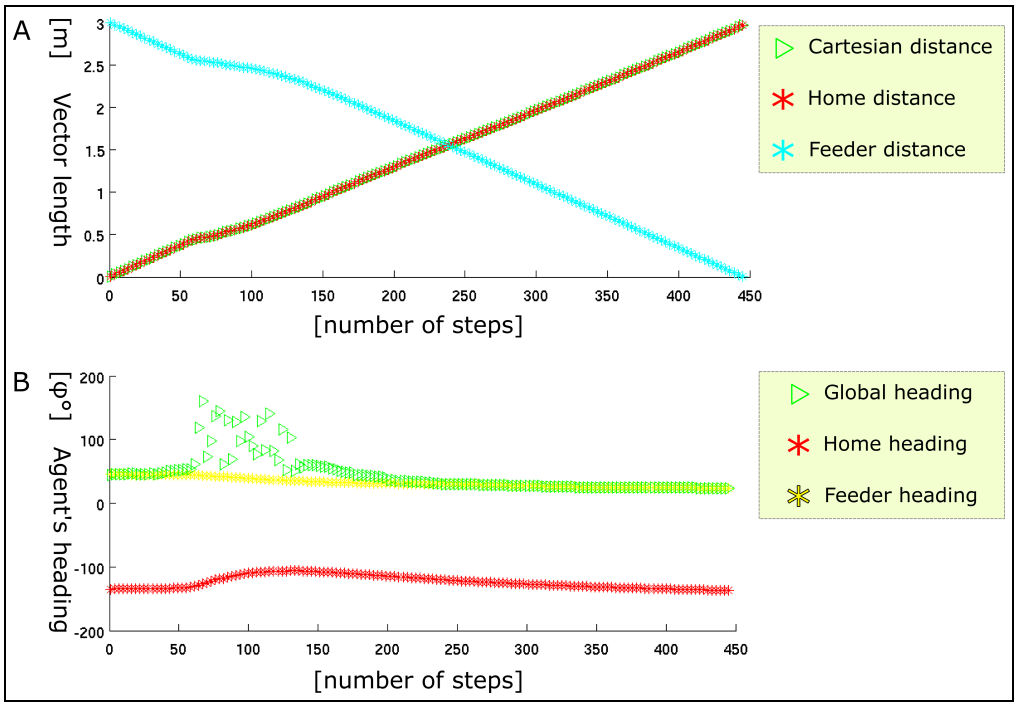


Figure 31: An illustration of changes in vector state in correlation to the avoidance of an obstacle. Figure 31A shows an overlap of both cartesian distance and the home/feeder distance as was computed by the path integration algorithm. The red/green curve starting at *vector length* = 0 demonstrates how the length of the home vector increased as the agent walked towards the feeder. The feeder curve (cyan) descends correspondingly, demonstrating how the length of the feeder vector was shortened as the agent got closer to the feeder. Both curves show a bend after around 60 steps, indicating a decrease in change of vector length related to the obstacle avoidance maneuver. Figure 31B demonstrates the angular change of the agent's heading in correlation with the obstacle avoidance maneuver. The agent was engaged in the avoidance maneuver starting at 60 steps after beginning of the experiment and ending it at around 140 steps from the beginning of the experiment. During this maneuver the headings of the feeder and home vectors were increased since the agent changed its course while avoiding the obstacle.

No change in the shape of the agent's trajectory or the distance from the obstacle could be measured comparing to figure 30 when the Gauss curve used as weight for the rejectors was changed from $\sigma^2 = 1.0$ to $\sigma^2 = 0.8$. Figure 33 illustrates the walking trajectory of the agent when using $\sigma^2 = 0.8$.

When the agent was tested with the 9-segments algorithm, some deviations from the trajectory of the 19-segments algorithm could be measured. The agent detected the obstacle first when it was 0.2m away from it (vs. 0.28m in the 19-segments algorithm). It then performed a 90^o turn to its left, completed a wider arc around the obstacle (compared to the agent in the 19-segments algorithm) and passed to its bottom left corner in a distance of 0.12m. For an illustration of the agent's trajectory using the 9-segment obstacle avoidance see figure 34.

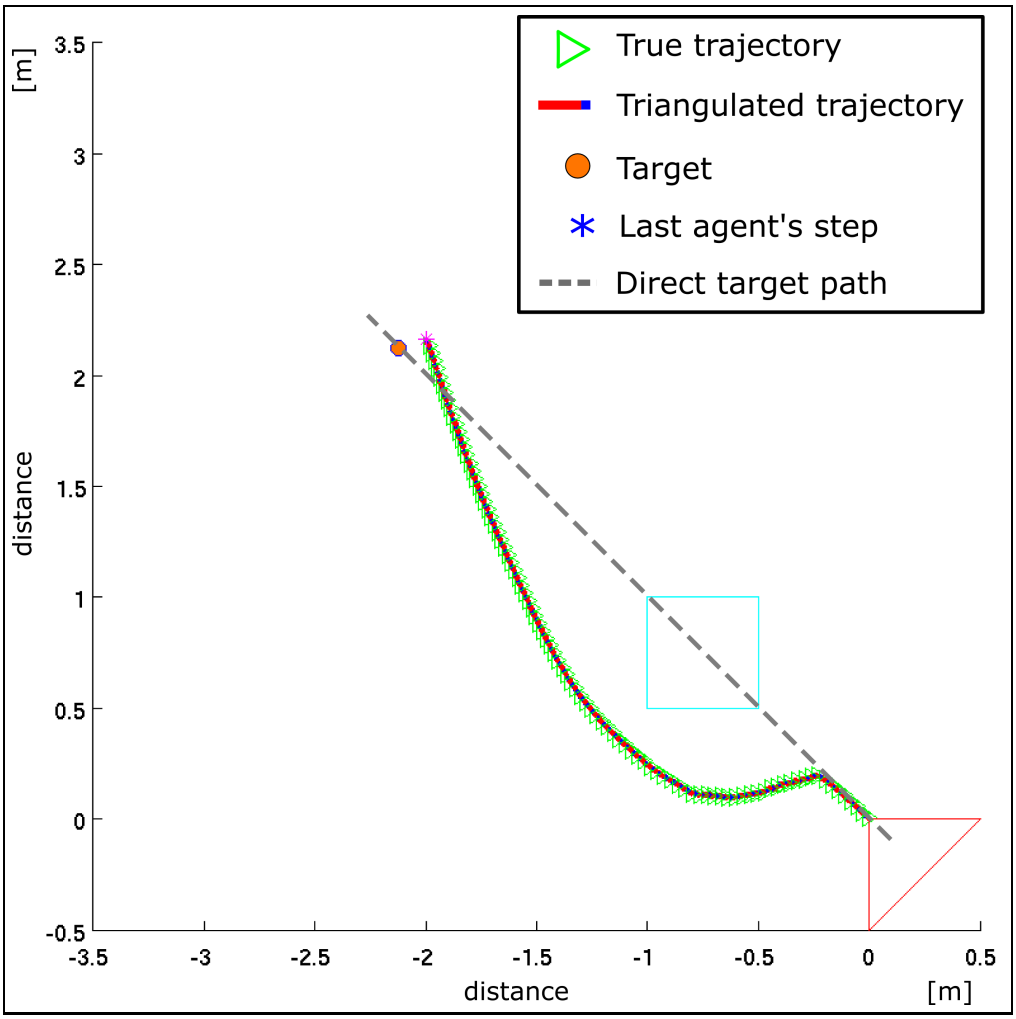


Figure 32: When the weight component μ adjusting the size of the target vector was reduced from $\mu = 0.04$ to $\mu = 0.02$ the agent maintained a bigger distance from the obstacle when compared to the previous experiment. All other agent and environment parameters were kept constant in this experiment.

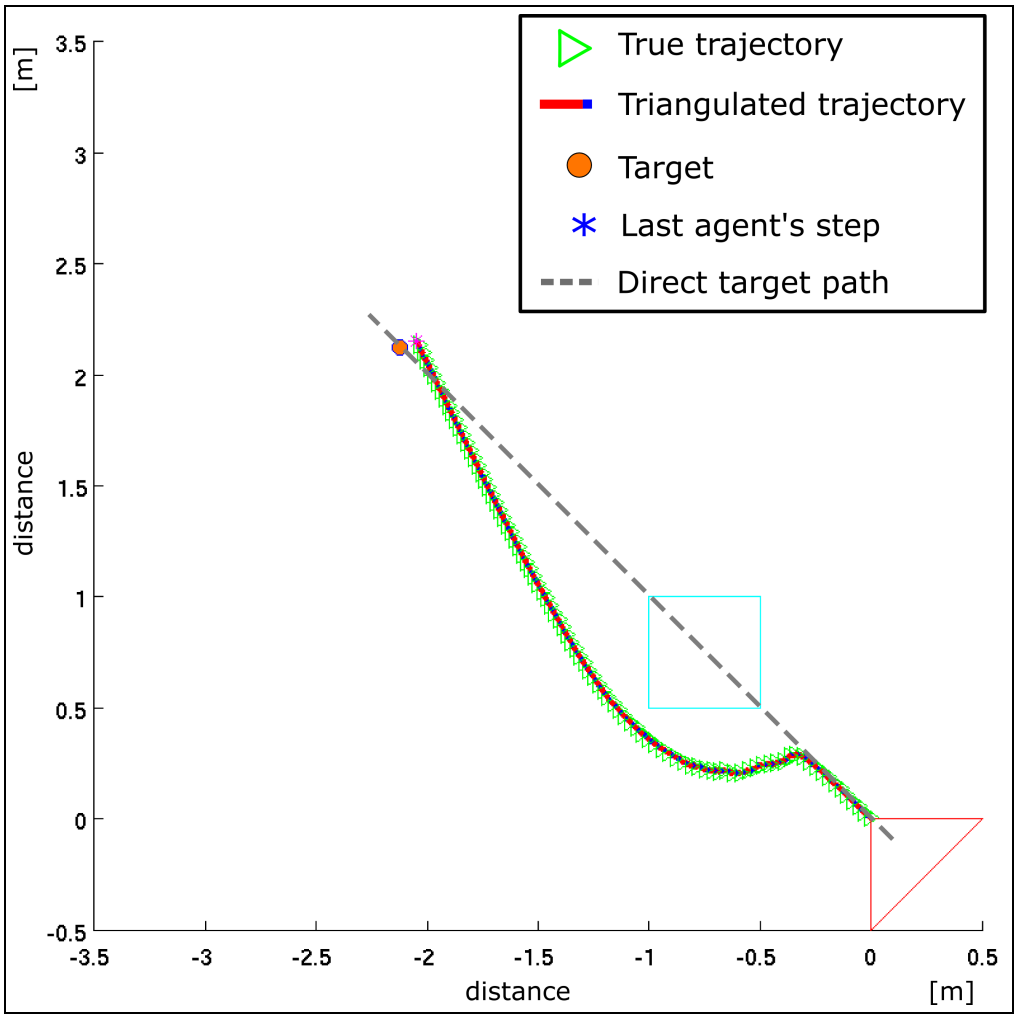


Figure 33: The agent's avoidance trajectory with a weight Gauss curve of $\sigma^2 = 0.8$. No difference to the agent's trajectory could be measured comparing to the trajectory of the agent with a standard weight cureve of $\sigma^2 = 1.0$. See figure 30 for comperison.

5.1.2 A non-symmetrically positioned obstacle

When the agent was tested in the environment with one non-symmetrically positioned obstacle, it turned to the opposite direction it turned in the environment with the symmetrical obstacle and avoided the obstacle. As figure 35 shows, the obstacle avoidance algorithm directed the agent to pass the obstacle on its right side.

5.2 Model validation based on behavioral data

5.2.1 The channel

The agent was observed to produce a centering response in reaction to different wall heights in the 3D environment. A centering response means that the agent fits its

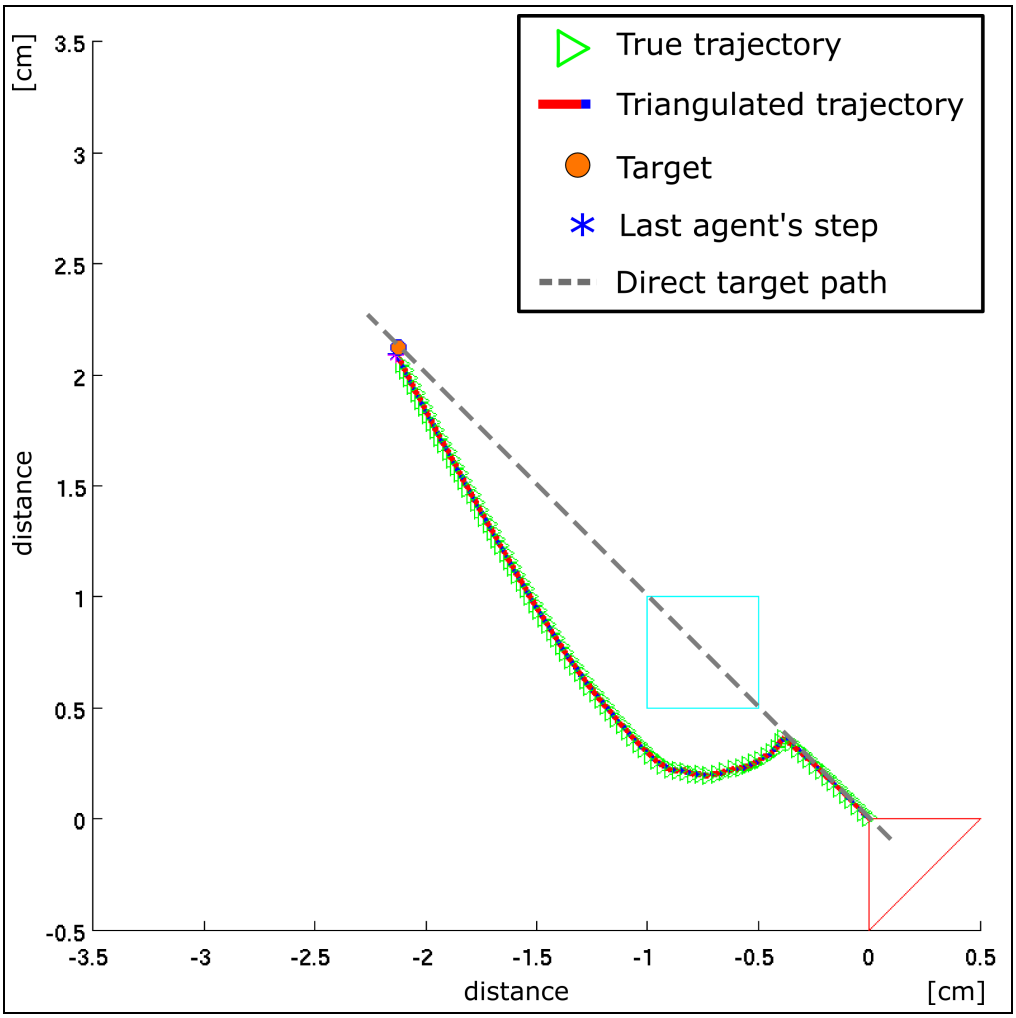


Figure 34: The agent's trajectory as was recorded when using the 9-segments obstacle avoidance algorithm. Some trajectory deviations comparing to tests of the same environment with the 19-segments (figure 30) algorithm could be measured. The agent reacted to the obstacle later (0.2m vs. 0.28m in the 19-segments algorithm) compared to the test with the 19-segments algorithm. The fact that the agent reaches the target this time is due to less acute triangles calclated by the path integration algorithm. Since less segments mean bigger angular values fed to the PIM, there were less errors calculated into the home and feeder vectors.

trajectory while walking in the channel in that way that obstacles (channel walls) appear to occupy the same visual sensor area. So in other words, the centering behavior refers to the visual egocentric perception of the agent. When both walls of the channel were of the same height (0.2m) the agent was observed to center its walking trajectory. The lateral distance of the walked trajectory from the channel walls amounts to 0.35m and is equal on each side of the channel's center line marked as gray dashed line. Figure 36 illustrates the walking trajectory of the agent through the channel with equally high

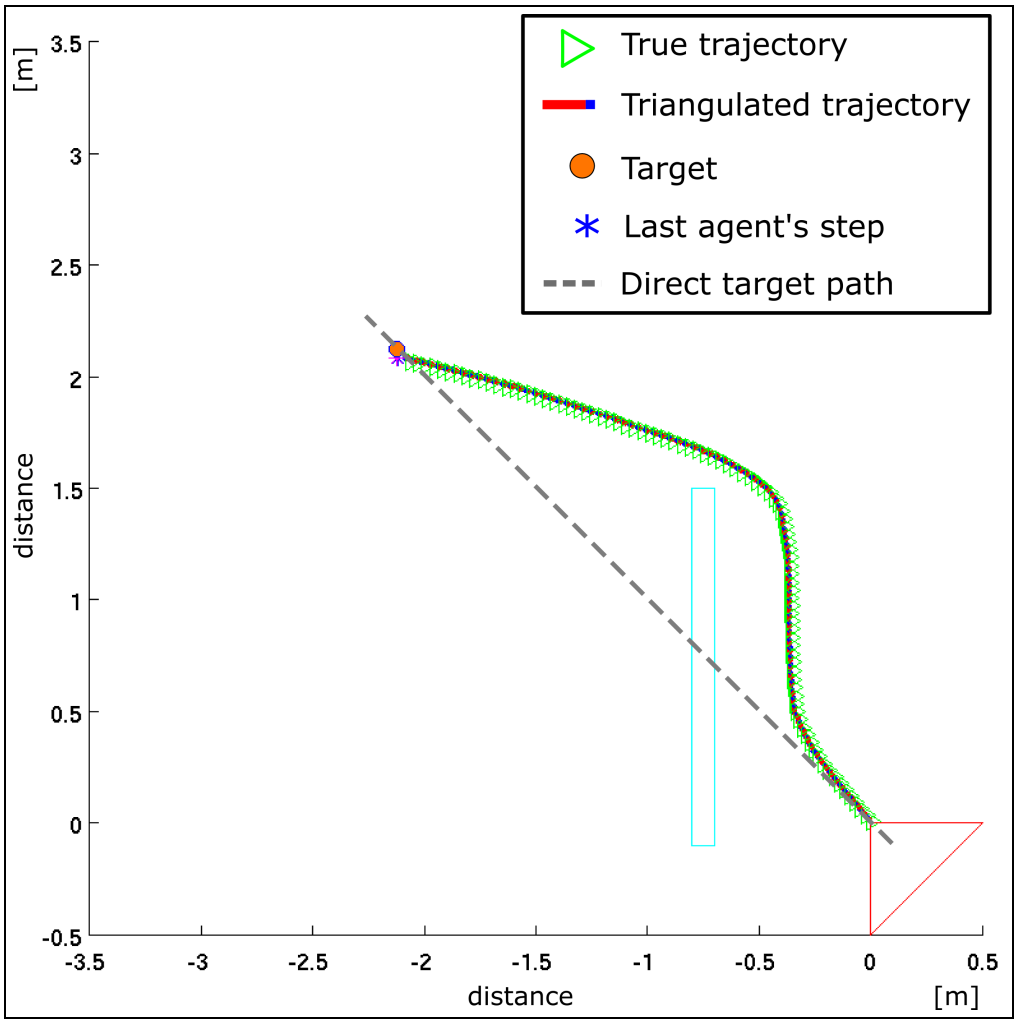


Figure 35: The agent avoided the obstacle on the right side (from the agent's point of view) while keeping a relatively big lateral gap between itself and the obstacle, comparing to previous experiments. The gap, in its widest spot, is equivalent to 0.3m. When turning into the direction of the target, after avoiding the obstacle, the agent passed the obstacle's corner at a distance that is equivalent to 0.1m in the real-world.

walls.

On the contrary, when experimenting with channel walls that were of different heights (0.2m left, 0.4m right) the agent changed its walking trajectory and walked closer to the lower wall. The agent was the closest to the channel wall after compleating a walking distance of 0.71m inside the channel. Its distance from the left wall was 0.30m. This meant an angular height of 33^o to the left and 45^o to the right wall, which was an angular ratio of $\frac{33^o}{45^o} = 73.3\%$. The angular ratio demonstrates how similar is the angular heights on each side of the agent. The walking trajectory of the agent in the channel with uneven wall heights can be seen in figure 37.

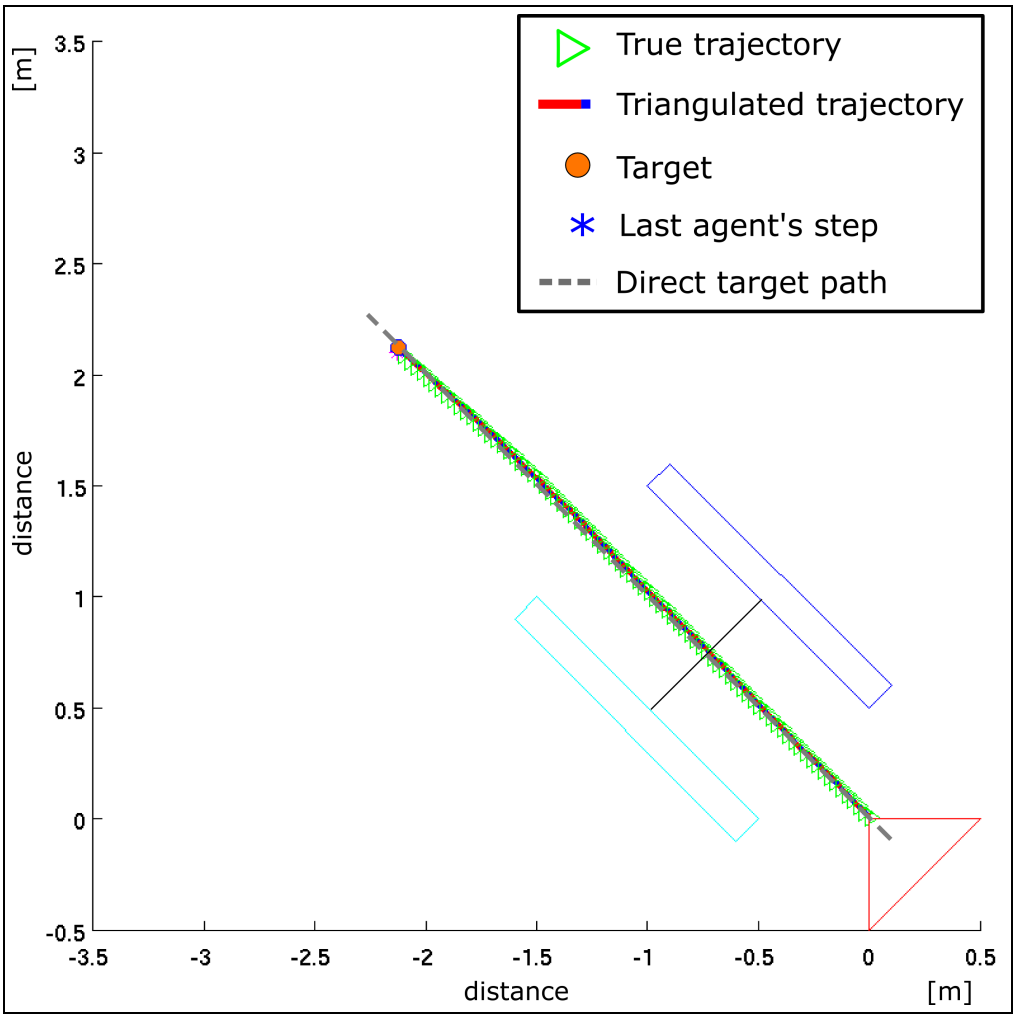


Figure 36: A trajectory of the agent walking through the channel with walls of the same height (0.2m). The trajectory shows a centering response of the agent in reaction to walls of equal height. The lateral distance from the walking trajectory to the walls was 0.35m on each side. The thin solid line perpendicular to the channel's walls designates the middle of the channel and the point of lateral distance measurement.

When μ, the target vector weight component, was reduced from $\mu = 0.04$ to $\mu = 0.02$ the agent's walking trajectory moved even closer to the lower (left) wall of the channel. After 0.71m of walking in the channel the agent's distance from the wall was reduced to 0.25m, which produced an angular value of 38^o to the left wall and 42^o to the right wall showing an angular ratio of $\frac{38^o}{42^o} = 90.4\%$. Figure 38 demonstrates the effect μ had on the walking trajectory of the agent through the channel.

When the height of the right wall was increased to three times (0.6m) the height of the left wall (0.2m) the agent's trajectory deviation from the center of the channel was even greater compared to both experiments introduced earlier (0.2-0.4m). Again, as in

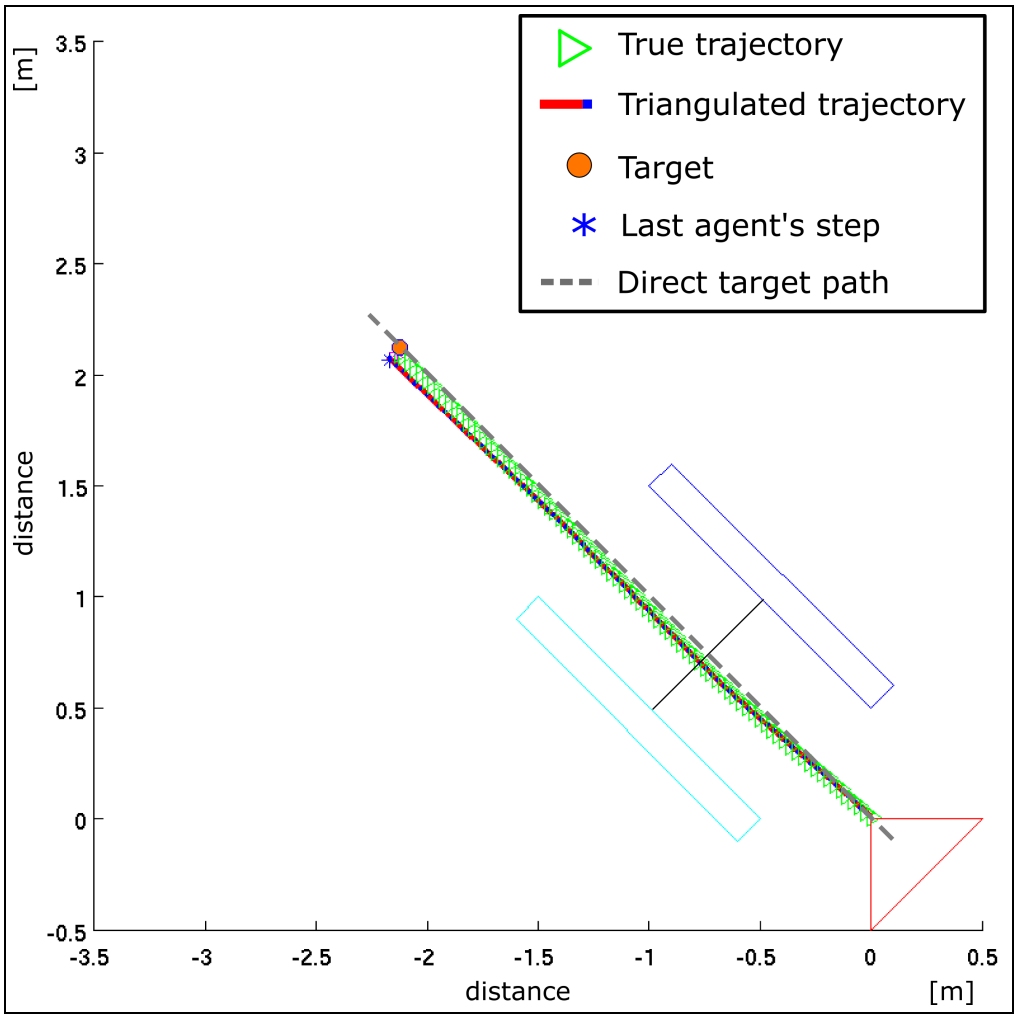


Figure 37: The agent's walking trajectory through the channel with walls of different heights. The left wall was 0.2m high and the right wall was 0.4m high. The distance of the agent to the left wall was 0.30m. Distances were measured at the location of most extreme deviation from the channel's center line. Measures were made at the point when the agent walked 0.71m into the channel. The thin solid line perpendicular to the channel's walls designates the middle of the channel and the point of lateral distance measurement.

the two previous experiments the agent reached the smallest distance to the left wall after walking 0.71m into the channel. The distance to the left wall was 0.22m meaning an angular height of 41^o to the left and 51^o to the right wall. This gave an angular ratio of $\frac{41^o}{51^o} = 80.0\%$. Figure 39 demonstrates the walking trajectory of the agent.

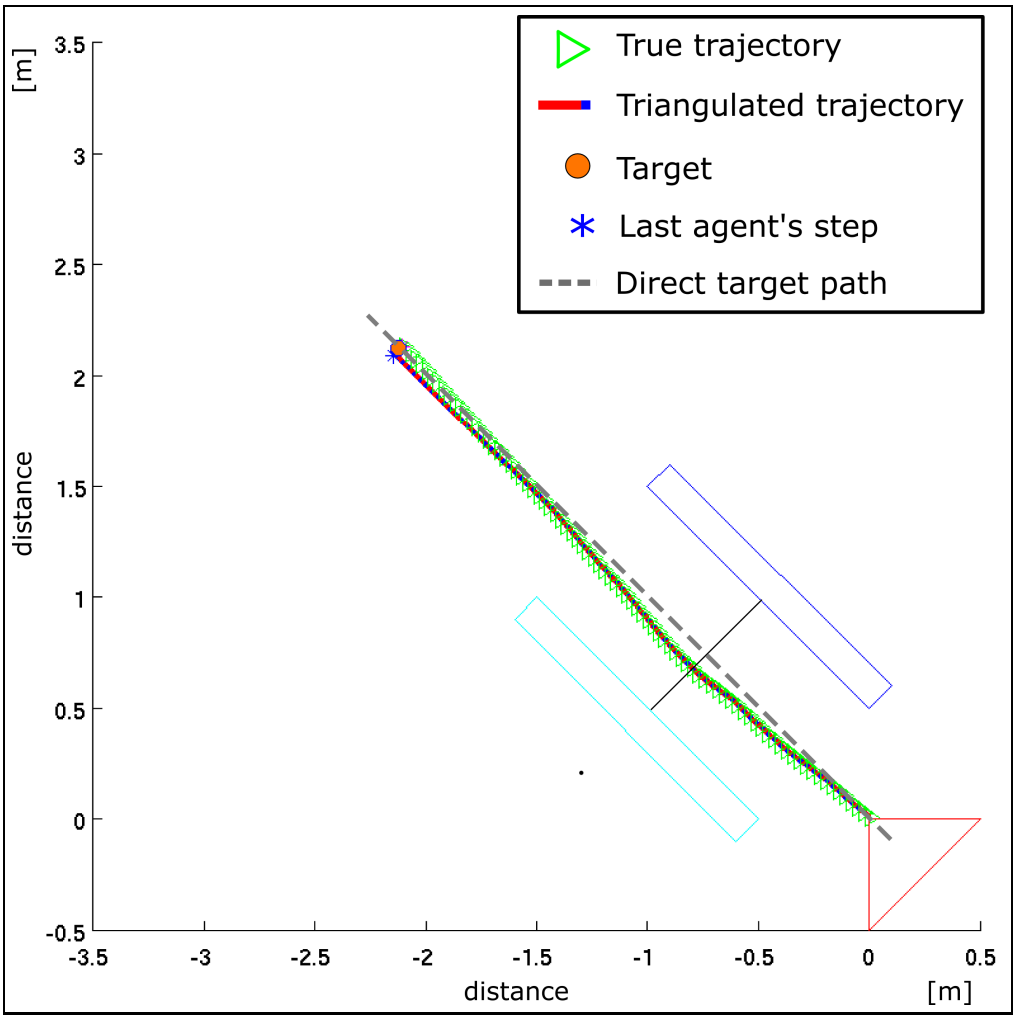


Figure 38: An illustration demonstrating the walking distance of the agent from the channel walls. In this experiment the target vector weight component was reduced from $\mu = 0.04$ to $\mu = 0.02$. This had an effect on the agent's lateral distance from the channel walls. After walking 0.71m in the channel the agent already reached the closest distance to the left wall measured in this experiment: 0.25m.

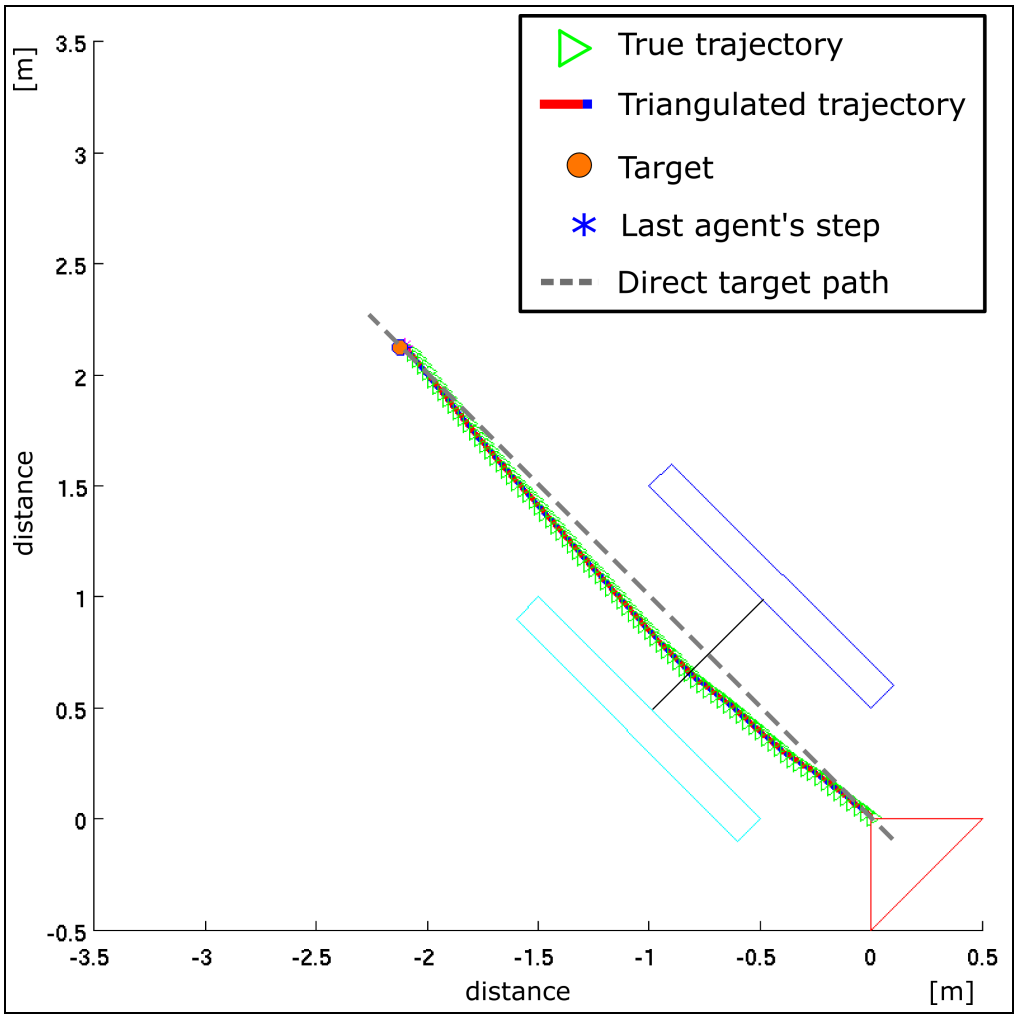


Figure 39: The agent's trajectory when walking through a channel with different wall heights. The right wall (0.6m) was three times higher than the left wall (0.2m). After walking 0.71m into the channel the agent reached the closest point to the left wall for this experiment: 0.22m.

5.2.2 The forced-detour paradigm

As illustraded by figure 40, the agent was able to avoid the forced-detour obstacle. Giving the agent the ability to avoid this obstacle was also the reason for the further development of the obstacle avoidance algorithm (see section 3.3.2).

Figure 41 shows the avoidance maneuver performed by the agent when still using the older version of the algorithm without attractors. After successfully avoiding the first (narrow) obstacle the agent was confronted with the second, wider, obstacle. Trapped in a local minimum behind it, the agent was unable to free itself and so it adopted an oscillating course walking back and forth along the wall of the second obstacle. Such a local minimum occurs when the driving forces moving the agent are similar from all

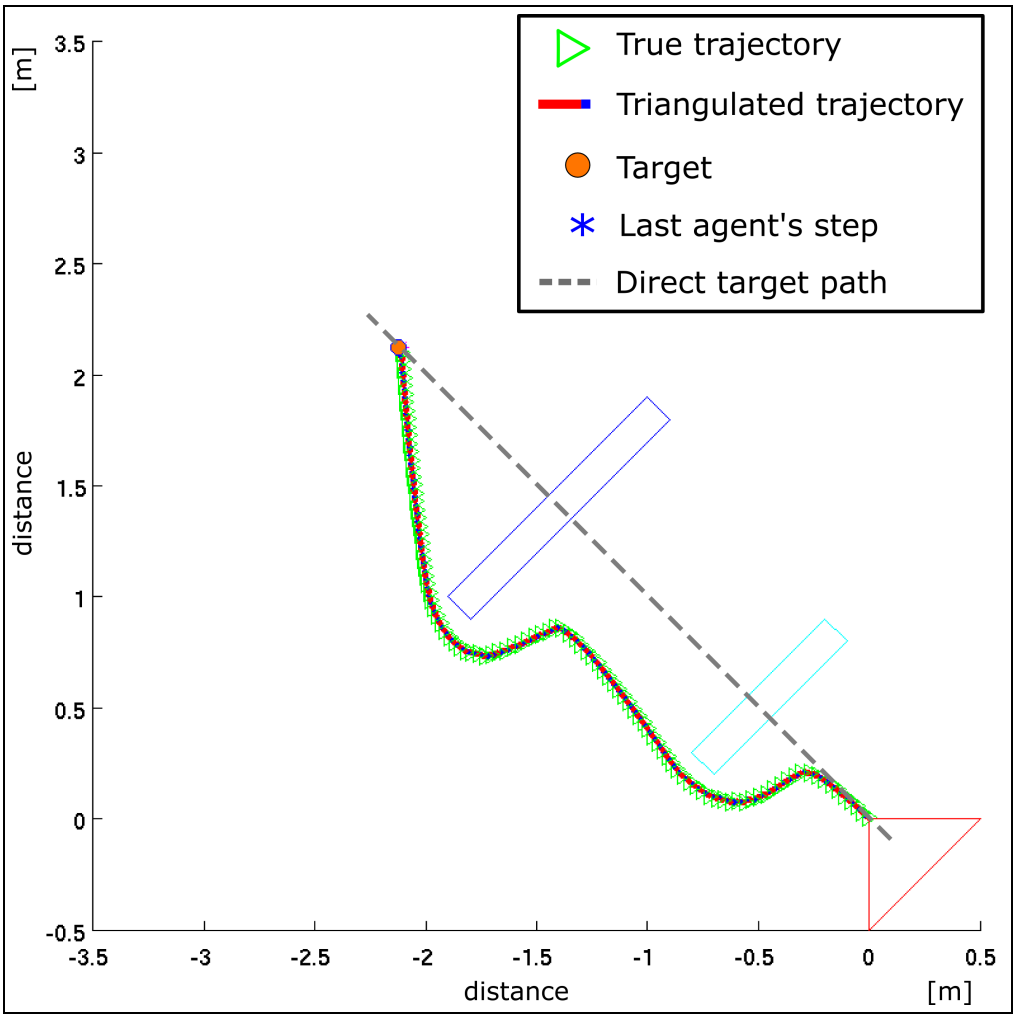


Figure 40: An illustration of the agent avoiding the forced-detour obstacle. This obstacle was the main reason for further development of the obstacle avoidance algorithm

directions, like a ball in a bathtub: all around it the ball encounters the same high rim, which overcomming it exceeds the current energy output of the ball and with no point in which less energy would have to be invested in order to escape the bathtub. The experiment was aborted after the agent performed around 700 non-productive steps, meaning it was deadlocked and was unable to pursue a course that would finally lead it to the target.

5.2.3 The cluttered environment

The walking behavior of the agent in the cluttered environment (described in sections 3.1 and 4.2.3) was tested two times, each with a different starting point but the same ending point. The walking trajectory of the agent in figure 42A, starting at (+0.3, +0.3), run to the right of the direction leading to the target (feeder). On the other

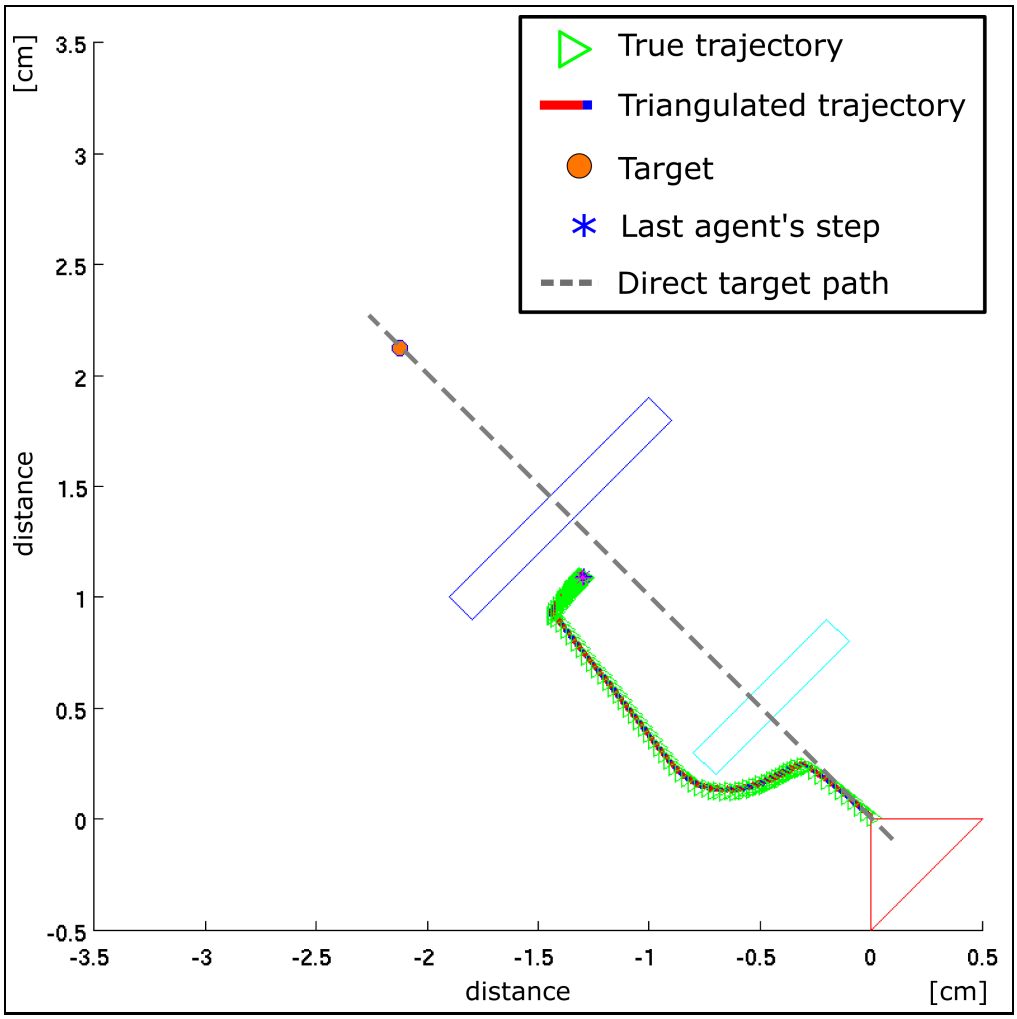


Figure 41: The agent encountered a local minimum while trying to avoid the second bar of the forced-detour obstacle. At this point it was unable to go around the obstacle and walked back and forth on a relatively narrow area. The experiment was aborted after around 700 non-productive steps taken by the agent.

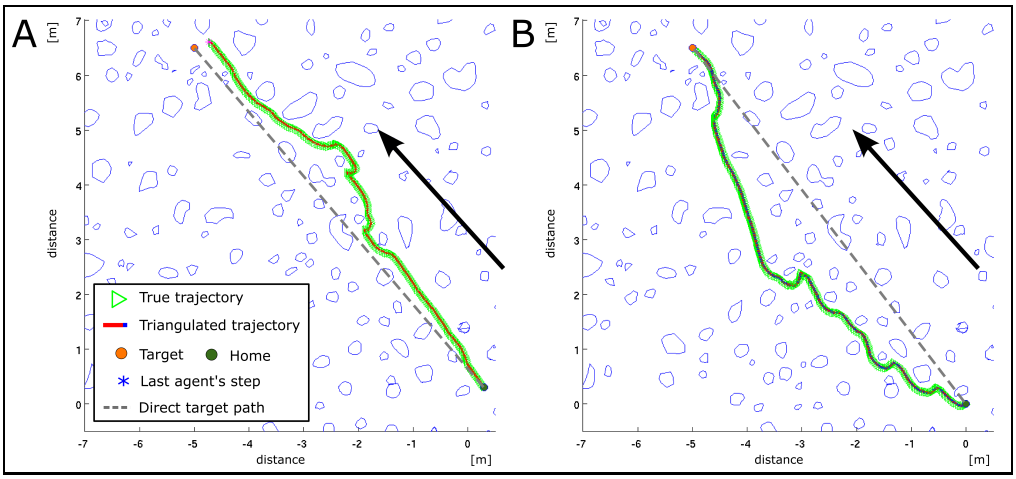


Figure 42: The trajectories of the agent through the cluttered environment. Each trajectory originated from a different location (+0.3, +0.3) and (0.0, 0.0). The end point was the same for both trajectories (-5.0, +6.5). The fact that the agent failed to reach the target is due to floating point rounding errors.

hand, the trajectory in figure 42B starting at (0.0, 0.0) run a wide arc to the left of the direct target line. In the cluttered environment the agent demonstrated movements between obstacles that were as close to each other as of circa 0.2m. This result could be varied to a certain extent by adjusting μ. The fact that the agent failed to reach the target is due to floating point rounding errors that fed the MOM with a slightly wrong target heading.

5.3 Deadlock environments

5.3.1 The infinite obstacle

The infinite wall presented the agent with an obstacle it could not possibly avoid. The obstacle was stretched over the whole testing ground (covered circa 90^o of the panorama to each side in target direction) and the agent was unable to go around it. Since the agent in the 3D environment was unable to climb on obstacles, the obstacle in figure 43 creates a local minimum in which the agent was trapped in. The experiment was aborted after it was observed that the agent did not manage to produce avoidance movements that would have eventually brought it closer to its goal.

5.3.2 The U-shape obstacle

The U-shape obstacle is a classic example for a local minimum situation. Figure 44 illustrates what happend when the agent encountered a local minimum while deep inside U-shape obstacle. In a distance of 0.47m into the U-shape the agent was trapped in a local minimum spinning around its own axis.

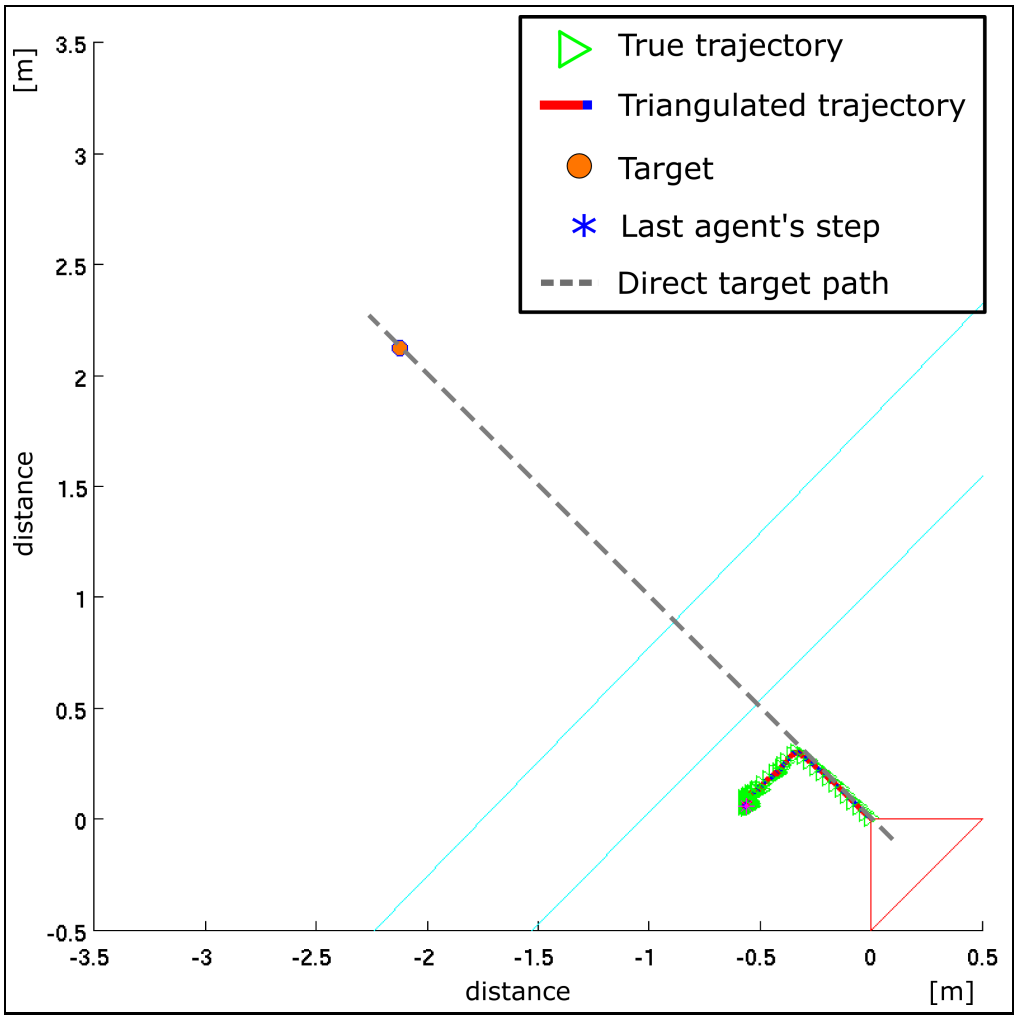


Figure 43: The agent faced a deadlock in the form of an infinite wall. After a rather short walking distance the agent turned to its left, walked along the wall and got trapped in a local minimum area and could not perform any further avoidance trajectory. The experiment was aborted after it was observed that the agent did not mange in making avoidance movements that eventually also brought it closer to its goal.

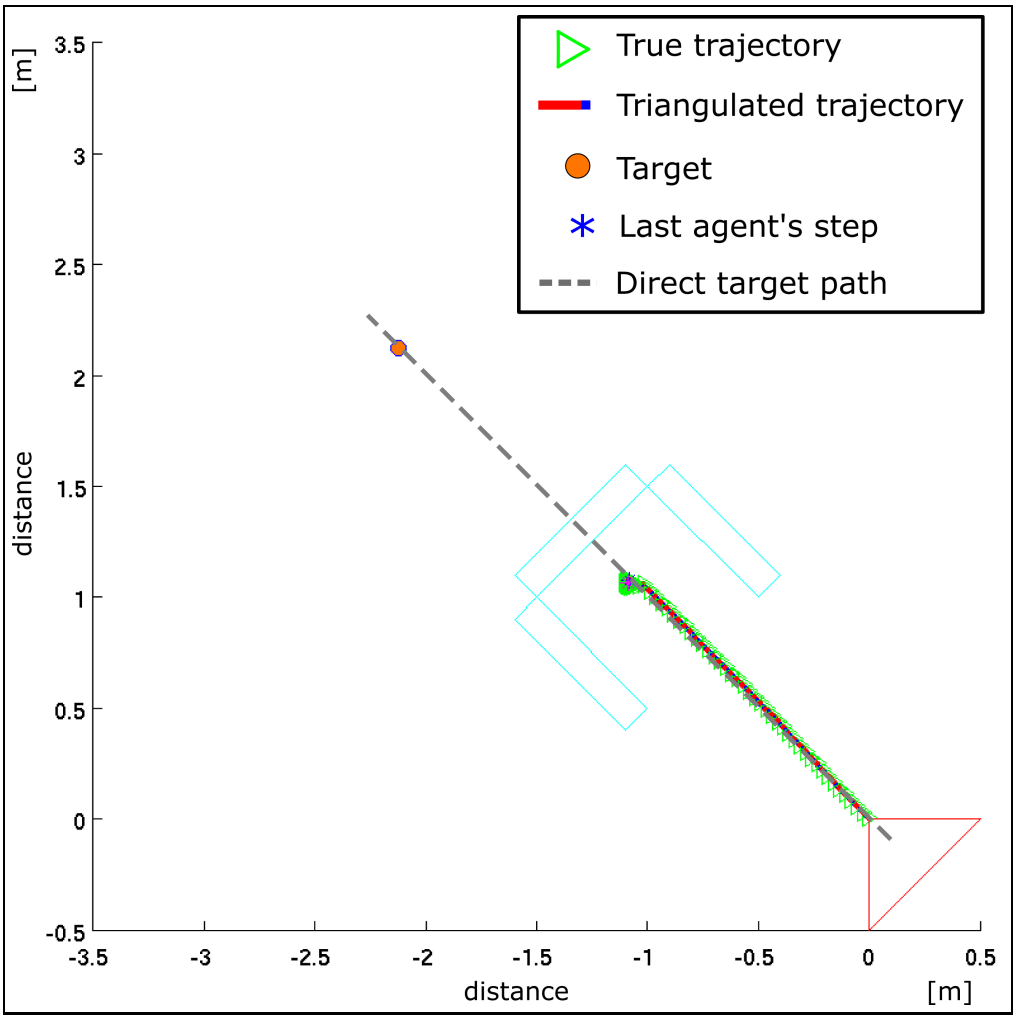


Figure 44: An illustration of the agent inside the U-shape obstacle. After walking 0.47m into the channel the agent was trapped in a local minimum. The experiment was aborted after it was observed that the agent did not mange in making avoidance movements that eventually also brought it closer to its goal.

5.4 Noise generator

Figure 45 illustrates the agent's walking trajectory when angular errors were integrated into the walking angular output. No comparison could be made with ant trajectories from field experiments, since there is no way of filtering out the errors the ants experiences when on foraging runs.

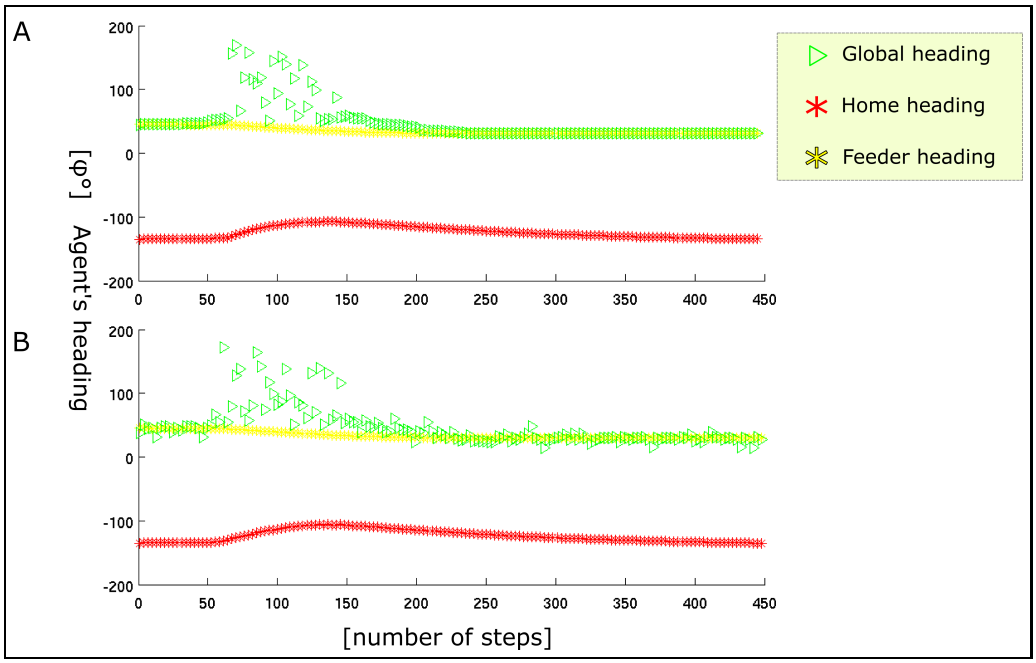


Figure 45: An illustration of the agent's walking path with and without angular error added to it. (A) No noise used. The global heading of the agent is kept steady when the agent is not avoiding the obstacle. (B) Angular noise is added. The agent's global heading oscillates as a result of adding randomized angular error to the angular movement output but no meaningful change in the agent's walking trajectory could be observed.

6 Discussion

In this work it was possible to demonstrate that a simulated model of a navigating ant could rely solely on real-time visual data for handling the task of avoiding obstacles. The model was constructed out of the three following modules: a *motor output module* in charge of moving the agent through the 3D-environment, a *vision module* in charge of detecting obstacles and producing avoidance angles and the *path integration module* in charge of keeping both home and feeder vectors updated. All three modules were essential for accomplishing the task of navigating towards a defined goals while avoiding obstacles positioned in the direct path of the agent.

6.1 The agent's parameters

Three parameters were manually adjusted in the 3D-model and are described in section 3.3.2: the number of segments dividing the agent's visual field, the weight given to the target vector and the weight given to each of the segments. Here I discuss the influence each of those parameters had on the agent's walking behavior.

6.1.1 The number of segments

Two versions of the model were tested, each with a different number of segments dividing the agent's vision field. The first version to be tested had 9-segments but later a 19-segments version was tested and used in the experiments. The number of segments was initially chosen so that each segment provided an opening angle that was equal to: $\frac{360^o}{9\ segments} = 40^o$. All but one experiment preseted here were run with the 19-segments version.

The difference between both versions was in the input resolution that led to a slightly different walking output. Since the angular value of each segment was measured from the center of the segment, increasing the number of the segments and thus implicitly decreasing the amount of angles each segment held, gave the agent a more accurate angular information regarding the location of the target and the obstacles. Indeed when comparing figure 30 (19-segments) and figure 34 (9-segments) in section 5.1, it is possible to notice that the agent adopted a smoother walking trajectory when using 19-segments. It started its avoidance turn earlier, turned to circa 130^o to the left instead of 90^o in the 9-segments system and completed a rounder arc around the obstacle. Since a change in agent walking trajectories could be observed after increasing the amount of segments, it is imperative to go a step further and increase the segment amount used by the obstacle avoidance algorithm. This will help define the threshold for segment count versus change in agent walking behavior. Unfortunately time limitations hindered this action.

6.1.2 Weight of target vector

The weight (μ) applied to the target vector had an influence on the force the target vector had on the agent compared to the force obstacles had on the agent. It was observed that when the target vector's weight was weakend from the standard value of $\mu = 0.04$ to $\mu = 0.02$ the agent kept a bigger distance to the obstacles. That can be explained by the rival vectorial force of the target versus the forces applied on the agent by the obstacles. Compare figures 30 and 32 in section 5.1 that show the difference in agent's walking trajectories. On the other hand, when the force of the target vector

was increased to $\mu = 0.05$ the agent showed a walking trajectory that was closer to the obstacles (not shown here). When set to a heigh value, μ caused the agent to walk through obstacles. In other words μ could also control the ability of the agent to successfully walk in a dense cluttered environment by allowing the agent to get closer to obstacles. It was observed that setting μ to a certain value would even help out the agent to free itself from regions of possible local minimum. See figure 46.

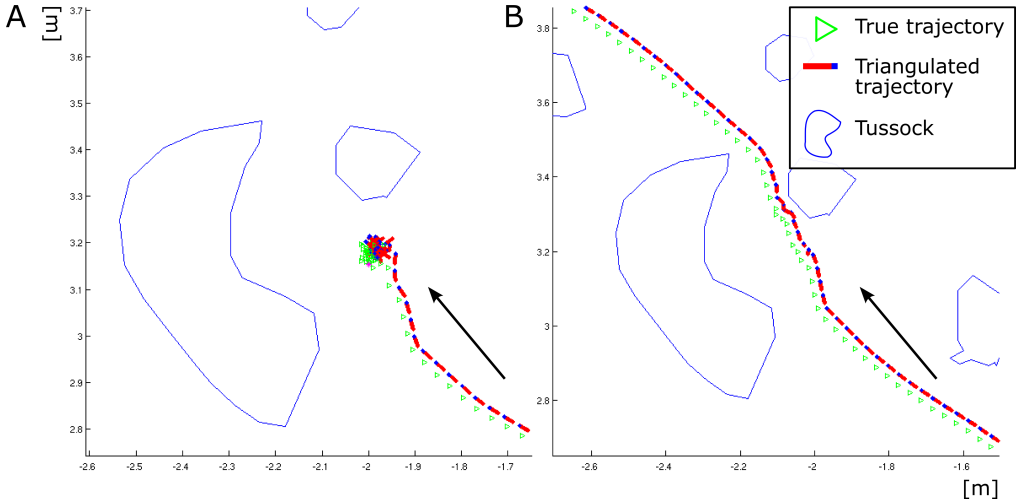


Figure 46: An illustration of how adjustment of the weight on the target vector can free the agent out of a deadlock situation. (A) $\mu = 0.04$, the agent is unable to avoid the two obstacles in its path and thus turns around its own axis. The obstacles create together a kind of U-shape since from the direction of the agent it is not possible to notice the gap between both obstacles. (B) Only after μ was changed to 0.05, the agent was able to get closer to the obstacles and 'notice' the gap between the obstacles. Then it also succeeded in avoiding the obstacles by passing between them on its way to the target. The blue dot on the triangulated trajectory represents the heading of the agent.

When decreasing the value of μ in the channel experiments (figure 38), the forces of the obstacles had more influence on the agent that walked closer to the lower wall compared to a previous experiment (figure 37) in which μ had the standard value of 0.04.
It can be postulated that such mechanism (if existing), could be adjusted by an ant for achieving a balance between saving energy and time (walk close to obstacles) and avoiding predators (stay away from obstacles).

6.1.3 Weight of segment vector

Weight was placed on each segment in the form of a Gaussian curve, according to the distance of each segment from the segment in which the target was located. For an illustration of curve shapes see figure 14. The weight defined which significance the agent gave to obstacles in its visual vicinity. It was more 'important' to avoid obstacles standing in the direction of the target than obstacles not in the agent's path to the target. When manipulating the variance σ^2 from 1.0 to 0.8, no changes in walking behavior of the agent could be observed. It was expected that the agent would react

earlier to an obstacle in the direction of the target since the factor $\sigma^2 = 0.8$ forms a more acute peak of the Gaussian curve. Accordingly it was expected that the agent would laterally pass obstacles closer compared to a segment weight of $\sigma^2 = 1.0$ since a factor of $\sigma^2 = 0.8$ creates a narrow curve, which midriffs rapidly descend. It is to assume that a reduction in the weight of the target vector, will make changes in the Gaussian weight noticeable. For results compare both figures 30 ($\sigma^2 = 1.0$) and 33 ($\sigma^2 = 0.8$).

6.2 The agent's behavior in the virtual environments

6.2.1 A symmetrically positioned obstacle

When approaching the obstacle, the forces pushing the agent away from the obstacle increased. This caused the agent to turn away from the obstacle and maneuver around it on its way to the target (figure 30). Because the obstacle is centered relatively to the agent's heading, the 'decision' of the agent to turn to the left or to the right of the obstacle must have been ruled by minute differences in pixel readings on each side of the panorama's center. See figure 47A for an illustration on how the agent viewed the symmetrically positioned obstacle in the 3D-world.

6.2.2 A non-symmetrically positioned obstacle

In the experiment of an obstacle that was positioned non-symmetrically according to the heading of the walking agent, the 'decision' in which direction to turn was more unambiguous. When the agent approached the obstacle, the left side of the obstacle was closer to the agent and appeared bigger on the agent's visual sensor. The right side of the obstacle was remote and accordingly appeared smaller on the visual sensor, thus the pixel count on the left side of the panorama was higher than on the right side, pushing the agent to the right. See figure 47B for an illustration of the agent's view field while approaching the non-symmetrically positioned obstacle. As the agent advanced along the wall, this scene was kept constant and 'pushed' the agent to the right edge of the elongated obstacle.

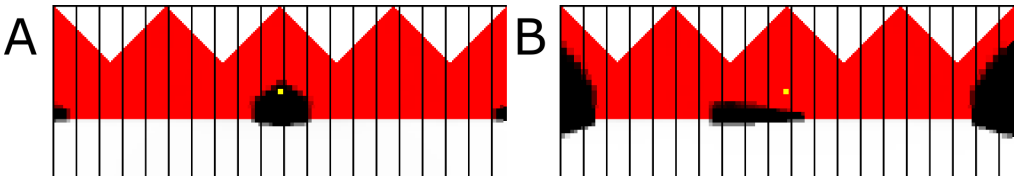


Figure 47: An illustration of how the 3D-world appears to the agent. (A) The symmetrically positioned obstacle. One can see that the obstacle covers almost the same sensor area on both sides of the panorama's center. (B) The non-symmetrically positioned obstacle. Altough the obstacle was uniformly high, the closest (left) part to the obstacle appears to be higher and covers more sensor area than the far away side (right) of the obstacle. The object on the edges of the panorama is an obstacle behind the agent.

6.2.3 The channel

Heusser and *Wehner* discovered that "*...ants of the species C. fortis always adjusted the lateral positions of their trajectories in such way that the two walls appeared to be of*

equal angular height" [11]. In their experiment (results shown in figure 48 on page 67) the ant's lateral positions within the channel coincided with the positions calculated for equal angular heights. Binomial tests, $P > 0.52$, Fig.48B, left; $P > 0.88$, Fig.48B, right. Although no significant results were received when measuring the deviation of the ant's trajectory from the line of equal angular heights, the deviation of the ant's trajectory from the midline was significant in both cases; $P < 0.001$ (marked in figure 48 as a red line).

As well as real world ants, the agent in the 3D-environment was observed to produce a centering response in reaction to different wall heights. In the case of the agent a centering response means that the agent fits its trajectory while walking in the channel so the obstacles (channel walls) appear to occupy the same visual field. Since the obstacle avoidance algorithm was only using the sum of pixels in each segment for calculating the rejecting force of the obstacles, its calculation relies only implicitly on the angular height of obstacles. When a high number of black pixels is registered in one segment, the chance for a high obstacle in this segment is bigger than in a segment with a small amount of black pixels but no actual 'angular verification' is run. The neuronal mechanism used by *C. fortis* to achieve this ability is unfortunately still unknown.

When both walls of the 3D-channel were of the same height (0.2m) the agent centered its walking trajectory because from the center of the channel the perceived size of each wall was equal. Figure 36 illustrates the walking trajectory of the agent through the channel with equally high walls. When the height of the right wall was doubled to 0.4m the agent's walking trajectory deviated from the center (0.35m from each wall) of the channel: 0.3m from the (lower) left wall. This meant an angular height of 33^o to the left and 45^o to the right wall. For an ideal view of equal angular elevation the agent had to walk in a distance of 0.23m from the lower wall, which would have produced an angular height of 41^o of both obstacles. By reducing the weight on the target vector from $\mu = 0.04$ to $\mu = 0.02$ an even better result was achieved. The produced angular value was than 38^o to the left wall and 42^o to the right wall showing an angular ratio of 90.4%. When testing the agent in a channel with a 0.2m high left wall and 0.6m high right wall, the agent walked at a distance of 0.22m from the left wall meaning an angular height of 41^o to the left and 51^o to the right wall. This gave an angular ratio of 80.0%.

Summing it all up, even if the agent did not present an ideal centring response to uneven wall heights, the results from the 3D-environment came close enough in showing a tendency. Considering the fact that the algorithm did not explicitly used angular heights for the calculation of obstacle rejecting forces, the results shown here are satisfactory. In the field experiment done by *Heusser* and *Wehner* the ants' walking trajectories were not significantly close to the lower wall. Only the deviation from the midline of the channel was significant. For a discussion on how the model could be adjusted to use angular information see section 6.4.3.

6.2.4 The forced-detour paradigm

With a minor change to the obstacle detection algorithm it was possible to make the agent capable of navigating around this obstacle that would otherwise trap it in a deadlock situation. By using *attractors* that pull the agent in the direction of segments that have a pixel count that is below a dynamically determined threshold, the agent was able to avoid the forced-detour obstacle and reach its target. The agent walked a trajectory that is akin to the trajectory recorded by *Wehner* [25]. Both virtual and real trajectories exhibit a zigzag walking pattern (when observing the total trajectory) in

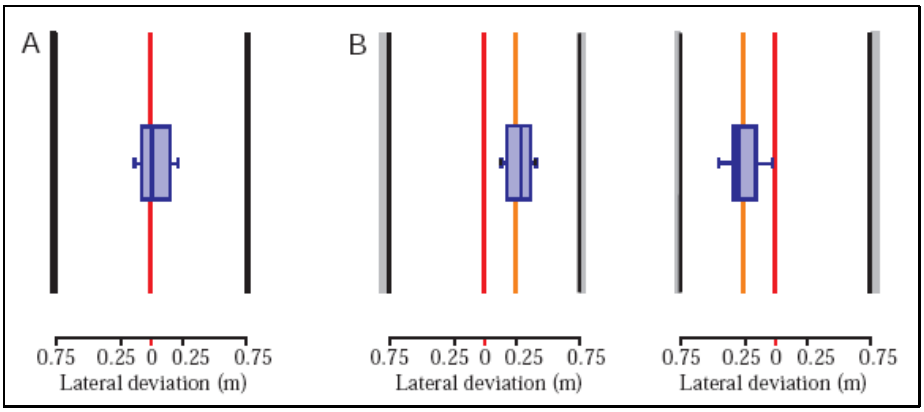


Figure 48: Results from the channel experiment run by *Heusser* and *Wehner*. (A) Both walls were uniformly black and of equal heights (0.2m) corresponding to 14.9^o coverage of the ant's eye as seen from the channel's midline (N=48). (B) either the left or right wall was increased to double the linear height (0.4m) corresponding to 28.1^0 as seen from the midline of the channel. The orange line is the position at which the walls appeared to project the same visual angle. The vertical blue line inside the box is the median of the ants' positions. The left and right boundaries of the box indicate the quartiles, which demarcate the range including 25% of the data to the left and right of the median. The horizontal bars indicate 75% range of the data, (left figure N=22, right figure N=21). Adopted from [11]

which both ant and agent are heading in the direction of the target when the obstacle is still too far to produce an avoidance reaction (like when walking between both obstacle bars).

Still, one cannot ignore the fact that the walked trajectories of the agent in the 3D-environment differ in a few points from the trajectories walked by *C. fortis* as described by *Wehner*[25]. In the trajectories shown by *Wehner*, the ant produced tortuous, rarely overlapping paths directly after leaving the nest, which became straight and overlapping when the ant reached the first obstacle wall. The agent in the 3D-environment did not produce tortuous trajectories when it was not engaged in an action of avoiding obstacles in its visual vicinity. It always walked in a direct and undisturbed line to the target and its trajectory was always the same one when tested in one defined environment. The reason for that is the determinism of computers, in which the outcome of their calculation is constant for repeating calculations with the same input. For a discussion about the integration of an noise generetor into the 3D-model see section 6.3. When further observing both model and real world trajectories, one notices that the ant in the real world tends to get very close to an obstacle's wall and walk along it whenever one exists. On the contrary, the agent's trajectory avoids the wall at any cost. It arches around the edges of the obstacle and in no point there is contact between the agent and the wall. Although *Heusser* and *Wehner* describe in a different experiment from 2002 that ants avoided the immediate neighborhood of (cylindrical) landmarks [11]. I know from personal experience that ants feel around walls and other obstacles with their antennas and front legs. In addition, figure 22 illustrates a walking trajectory that is tightly close to the walls shown in the experiment. For a discussion about the property of ants to follow walls and how it could make the model presented here more complete

see section 6.4.1. Even though the descriptive and quantitative data known about the forced-detour is limited, the ant's trajectories seen in figure 22 reveal one further detail about navigation properties in ants: one ant trajectory can be seen crossing the dotted line represending the direct route to the target and the median from which both obstacles are evenly divided. The ant than decided to turn around and avoid the wall on a path that was now longer than if it would have kept on going. Such walking behavior is not possible in the 3D-model. Since the target vector (home or feeder) is tightly embedded into the walking behavior of the agent, it has only two possible choices when encountering a similar situation:

- Either the wall will be too long and will create a local minimum, from which the agent could not free itself.
- Or it will go on with its trajectory and avoid the obstacle on its left side (reader's perspective).

Whenever the agent crosses the midline to the target, its target vector crosses the midline too, keeping the agent on that side of the obstacle. Ants, as it seems, use their target vector in a more flexible way, which enables them to perform scouting and searching trajectories more dependent on the features of the terrain than on the constrains of their target vector.

6.2.5 The cluttered environment

The agent was able to successfully navigate through the cluttered environment and reach its target, while passing between narrow gaps formed by the tussocks. It could be shown that by adjusting the target vector weight, the agent can be made to free itself from deadlock situations. Further, it was shown that, even when using one algorithm to solve all obstacle avoidance tasks, a minor change in the location of the starting point, can have a major effect on the walking trajectory of the agent. When the agent was advancing through the cluttered environment, each of its 'decisions' regarding the avoidance of a certain obstacle, served as a further link in a chain reaction that had immediate influence on its walked trajectory. Still the agent's walked trajectory and the original ant's walked trajectory cannot be simply compared. Ants walked trajectories are influenced by further factors than only the size of obstacle in their visual vicinity. Visual landmarks as well as olfactory landmarks play an important role in the navigation abilities of real-world ants. This fact distinguishes the multi-factored nature of the real-world from the sterile, low-factored simulated world. Figure 42 shows how small changes in the beginning of the foraging run, dramatically change the ant's walking trajectory.

6.2.6 The infinite obstacle

The infinite wall presented the agent with an obstacle it could not avoid. Since the wall was as long as the agent could see, it trapped the agent in a local minimum. The agent than moved back and forth between two points along the wall. For a possible solution to the local minimum problem see section 6.4.1. It could be interesting to run similar experiments in real-world ants that will show, which strategies ants use in order to overcome such obstacles. Will a foraging ant on its way to a known feeding point give up its efforts to reach that feeding point once a very long obstacle is suddenly placed between it and the feeding point? How long must such an obstacle be for an ant to give up on a profitable feeding spot? How much time will an ant be willing to invest in

such an avoidance maneuver? If an ant does finally give up trying to avoid an obstacle, that could mean that ants probably possess a some kind of mechanism that constantly examines the effort to reward ratio of a certain path. Does ants behave differently when confronted with the same obstacle but on different sides of the foraging run (taken than the obstacle is symmetrical)? Will an ant invest more efforts in trying to overcome an obstacle on its way to the nest than on its way to a known feeding ground? All these still unanswered questions, show how little is known about obstacle avoidance in desert ants.

6.2.7 The U-shape obstacle

The U-shape wall produced equal rejecting forces from three sides around the agent. This caused the agent to be trapped in a local minimum that was even tighter than the local minimum created by the infinite wall. This was demonstrated by the walking pattern of the agent that consisted of turning around itself after entering the local minimum zone. The force pulling the agent towards the target (and the central U-shape wall) prevented the agent from moving back into the direction of the opening in the U-shape, which would have been the 'correct' solution and so it kept on turning around its own axis.

6.3 Noise generator

Although noise was added to the motor output of the agent, no extensive change in walking behavior could be detected. When comparing two walking trajectories in the cluttered environment, once with and once without noise, no changes in the agent's 'decision making' was registered (not shown here). Eventhough there were some minute changes in walking position relative to the tussocks, the agent passed all tussocks in the test with noise added, on the same side as with the experiment without noise. In the desert ant *C. fortis* the situation is similar. As *Wehner* writes: "*...The ant's path integration system yields relatively large errors only when the ant performs sharp backward turns...*" but goes on to explain that such turns rarely occur [16]. A further factor is the fact that errors often cancel each other because of the even distribution of the related errors and the fact that *C. fortis* turns as often to the right as to the left, so that an overall directional bias is unlikely to develop.

6.4 Improving the model

The 3D-model presented here is by far not a complete one. As was demonstrated by the *infinite* and *U-shape* environments, there are situations in which the agent gets trapped in a local minimum and is not able to reach the target. In order for the agent to be able to free itself, the obstacle avoidance algorithm should be extended to include non-linear solutions for dealing with specific deadlock situations. By using the phrase "non-linear" it is meant that the reaction of the agent to different obstacle constellations will not be steady, so certain situations might invoke different methods of obstacle avoidance. Those possible methods should be presented below.

6.4.1 Follow the wall

In the 3D-environment the agent encountered two situations (see section 4.3), which created a local minimum trap. As currently designed, the obstacle avoidance algorithm

implemented in the 3D-model is unable to offer a suitable solution for cases in which rejecting forces coming from different obstacles hinder the agent on its way to the target. One possible method used in robots to avoid a local minimum situation, is the wall-following method presented by *Koren* and *Borenstein*[3]. In their work, *Borenstein* and *Koren* implemented in a moving robot a non-linear algorithm for avoiding obstacles that used two different avoidance methods while navigating. If the robot was heading in the direction of the target, it used a virtual force field (VFF) that rejected it away from obstacles blocking its path. On the other hand, if the robot was heading into a direction, which was higher then 90^o compared to the target, that meant that the robot was moving away from the target and could get trapped in a local minimum. In such a case a wall following mechanism was activated and led the robot along a wall until its heading decreased below the 90^o threshold. Then the robot switched to VFF mode once again. Ants and other arthropods like bees and wasps are also known to follow walls or other contour lines while navigating [19]. For worker ants, which are unable to fly above obstacles, this ability to follow the outline of an obstacle is essential for reaching their goal. When unable to climb over an obstacle or fly above it, walking along the outer line of an obstacle is likely to lead the ant to an opening where it could regain its target's path. Even inside an arena, which represents the most extreme form of obstacle, walking along the inner arena wall might eventually lead to an opening and be more effective than a random walk or concentrating solely on the region pointing in the direction of the target. Unfortunately little is known about the methods ants apply in order to avoid obstacles and which effect those methods have on the shape of the walked trajectory.

In the forced-detour paradigm (section 4.2.2) *C. fortis* allowed science a small glimpse into the behavior of ants regarding a certain obstacle constellation (two parallel walls) but there is no quantitative and too little qualitative information supplied about this experiment. It would have been interesting to know how far from the wall ants changed their trajectory to avoid hitting the aluminum bar and whether ants would walk all the way along the metal bar and avoid the metal bar on its right side.

Combining the current implemented algorithm with a wall following algorithm that would be activated when certain situations occur, will definitely add to the robustness of the presented model. See section 7.1 about possible future experiments that might shed light on the way desert ants cope with different constellations of obstacles.

6.4.2 Assigning a secondary target

The idea behind following a wall is actually creating a secondary target and giving it temporarily a higher priority compared to the primary one. This secondary target is created whenever a possible deadlock situation is detected and is active until the path to the target is cleared. A possible way of detecting potential deadlocks is by judging the angular size an obstacle occupies in the panorama or by watching the heading of the agent compared to the direction of the target as described by *Borenstein* and *Koren* in [3]. A further mechanism would be to implement a walking memory that will monitor the length of the target vector over time. If the vector's length stays constant for a certain time than a deadlock situation has occurred.

When an ant is confronted with a big obstacle, the primary objective of reaching its goal must be temporarily put aside. The new objective that emerges is to avoid the obstacle but still keep the primary objective in memory for otherwise the target can not be reached.

6.4.3 Using angular visual input

It is mostly unknown how angular visual information is integrated into the navigating behavior of ants. Considering the agent, the solution is a straight forward one. By increasing the number of segments to the number of the panorama's width in pixels, each segment will become 1 pixel wide. Counting the pixels in each segment will indirectly serve as a vertical value for obstacle presence in those segments. It is possible that obstacle pixels will be concentrated in the top of segments, thus allowing free passage of the agent underneath, but such situations can also be dealt with, for example by dividing the processing of each segment into a lower and upper parts. Segments with full upper parts but empty lower parts could than be classified as passable segments.

Such modification to the obstacle avoidance algorithm would also supply a much more detailed information about obstacles in the environment. This detailed information might also be used as a 'low level' object differentiating mechanism. This could be done by identifying the gap between obstacles in segments that have no black pixels in them.

6.4.4 Obstacle distance

No measures were taken in enabling the agent to estimate the distance to obstacles in its visual vicinity. Such ability would make it possible for the agent to distinguish different obstacles by their spacial depth and detect passages between obstacles normally hidden from its visual field. Such an ability could also be used in a priority system that gives closer obstacles higher avoidance urgency then far-away ones.
Two methods could be implemented in the model to supply the agent with depth information:

- Distance estimation by Optic flow. Close objects produce a fast changing optic flow compared to objects in the distance. By analyzing the change in optic flow patterns, it could be possible to give the agent the ability to recognize different obstacle depths as they appear in its visual field. It is known that desert ants use self-induced ventral optic flow to estimate their walked distance [20] but whether this ability can also be used to estimate the distance to a certain object, remains unknown.

- Distance estimation by the angular declination below the horizon. Humans are able to use the angular declination below the horizon for distance judgment [18]. Ooi et. al (2001) found (in humans) that "*...when the angular declination was increased by binocularly viewing through base-up prisms, the observer underestimated distance. After adapting to the same prisms, however, the observer overestimates distance on prism removal...*". Again, this ability was never tested for in desert ants. One difficulty for ants to estimate distance in such way is their very small distance from the ground, since their visual angle under the horizon is very limited. Still, one good candidate if any, to use such means of estimating distance to objects in its visual vicinity could be *Cataglyphis*, since it is certainly one of the tallest of ants known to science (circa 1cm away from the ground). Regarding the agent, it would be possible to detect the first appearance of obstacle pixels from the bottom of the 3D-floor and thus the distance of an obstacle from the agent.

6.4.5 Model accuracy

As mentioned earlier in section 3.3.1 on page 25 there was a problem with calculations of floating point numbers. Those calculations produced slight numerical deviations that

accumulated to noticeable deviations of the walking path from the target. Unfortunately this fact was discovered too late for action to be taken. Here I shall offer two different solutions for working around this problem.

The first solution would be to multiply each floating value with a very big number, e.g. 10^8. Such multiplication shifts eight digits to the left of the decimal comma and increases the accuracy of such calculation. After all calculations are done, the result must be reduced to the original decimal state of the initial values.

Another method would be to utilize a different triangulation formula. Instead of using the error-prone *arccosine* equation appearing on page 25, the more accurate *arctangent* equation presented below should be used:

$$\beta = arctan \frac{\overline{AC} sin\gamma}{\overline{AB} - \overline{AC} cos\gamma} \tag{12}$$

7 Outlook

Further work should be done in the field of obstacle avoidance in desert ants if a more complete model is to be constructed. This work could be divided into two branches in which the first one is intensive field work experimenting with obstacle avoidance in real-world ants and the second one is further construction of the model based on new data gathered in field work.

7.1 Future experiments

7.1.1 The ring obstacle

No scientific work dealing with the reaction of desert ants to an infinite obstacle was ever done. In this experiment an ant will be trained to walk back and forth between a nest and a feeding location. After the ant will learn the route from the nest to the feeder and back, a 0.2m high and 1m in diameter plastic ring will be placed around it while it is on its way back to the nest. Since the ring blocks the ant from all sides, it represents an infinite obstacle the ant cannot avoid. The interesting question is how will the ant react. What will be its walking trajectory? Will the ant change its avoidance behavior as time advances? Figure 49 demonstrates a possible outcome of such a ring obstacle experiment.

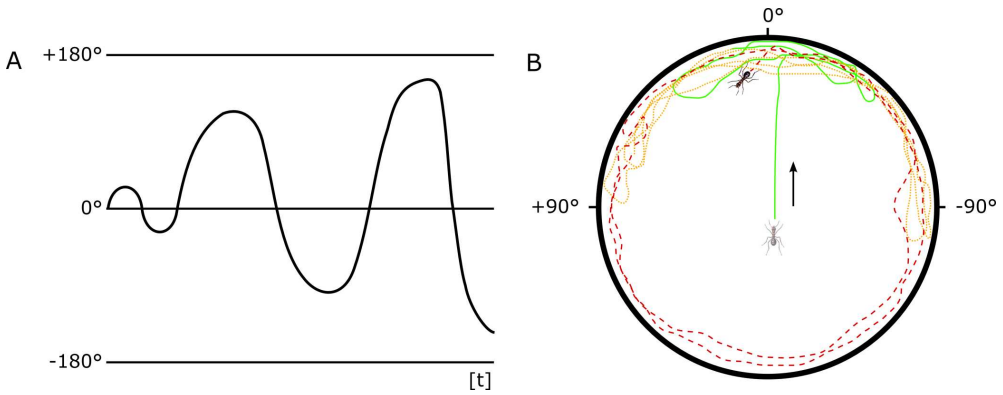


Figure 49: The ring experiment for suggested future work. (A) The angular walking trajectory of the agent over time. The assumption is that the agent will oscillate inside the ring. first it will attempt to avoid the obstacle by making small maneuvers expanding those maneuvers as time advances and no voidance solution is found. (B) A schematic walking trajectory of the agent as seen from a bird's eye view. The low contrast ant image marks the beginning of the trajectory. The azimuth 0^o is pointing to the nest. The fully colored ant represents the last step taken by the ant. The solid green trail is the beginning of the ant's attempts to avoid the obstacle. This trail limits itself to a relative narrow angular area. The dotted orange line illustrates the ant's trail in a later time followed by the dashed red line representing a full inner circle of the ring.

7.1.2 A channel with varying wall heights

How would an ant react to channels with varying wall heights? How will its walking trajectory look like when the ant walks in a channel in which one wall becomes lower as

the ant advances through the channel. Will the ant get closer and closer to the lower wall, dynamically adopting its visual angular perception of the obstacle? According to the work by *Heusser* and *Wehner* mentioned above [11] it is plausible to assume that ants tested in such an experimental environment will get closer and closer to the lower wall, in attempt to balance the vertical angular size of the walls around it.

Such experiments could shed light on the way ants cope with different kinds of obstacles and contribute to the 3D-model by suppling relevant real-world data.

7.2 Further model development

As mentioned in section 6.4, several improvements could be integrated into the 3D-model that would enhance its ability to simulate the obstacle avoidance behavior of the ant in an even more accurate manner. Further development could include the use of computer neuronal networks to more accurately simulate the decision making process in the agent. Further it could be experimented with overlapping receptive fields (segments) as they are known from some invertebrates and vertebrates for creating a more robust vision mechanism.

8 Acknowledgments

No meaningful path is ever paved by oneself.

For that I would like to mention all those who were willing to give of their time and effort to make this work become what it is.
Many thanks to Professor Dr. *H.A. Mallot* for putting me on the right track, thanks to *Yossi Yovel* for his meaningful remarks and mathematical brains, thanks to *Matthias Wenzel* for the efficient brain storms, thanks to *Axel Rezlaff* for the very usefull remarks, thanks to *Holli Mathis-Masury* for her professional advice in matters of the English language, thanks to *Avi Mathis-Masury* for showing me things from a different perspective, thanks to *Andrea Kramer* for standing me during this period and for pulling me up whenever I felt like being down, many thanks to *Kai Basten* for his endless patience, for his ability to clearly explain complex matters and for letting me access to his technical knowledge in the fields of computer science and ant research.

And last but not least, many thanks to *Cataglyphis* and her close relatives for only allowing us bit by bit insight into their mysterious and fascinating world.

References

[1] S. Åkesson and R. Wehner. Visual navigation in desert ants *Cataglyphis fortis*: are snapshots coupled to a celestial system of reference? *J. Exp. Biol.*, 205:1971–1978, 2002.

[2] A. N. Andersen. Common names for australian ants (Hymenoptera: Formicidae). *Australian Journal of Entomology*, 41:285–293, 2002.

[3] J. Borenstein and Y. Koren. Real-time obstacle avoidance for fast mobile robots. *IEEE Transaction on systems, Man and Cybernetics*, 19:1179–1187, 1989.

[4] Xim Cerdà. Behavioural and physiological traits to thermal stress tolerance in two spanish desert ants. *Etologìa*, 9:15–27, 2001.

[5] L. Chittka. Optimal sets of color receptors and color opponent systems for coding of natural objects in insect vision. *The Journal of Theoretical Biology*, 181:179–196, 1996.

[6] M. Collett and T.S. Collett. How do insects use path integration for their navigation? *Biol. Cybern.*, 83:245–259, 2000.

[7] T.S. Collett and M. Collett. Path integration in insects. *Current Opinion in Neurobiology*, 10:757–762, 2000.

[8] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction To Algorithms*. The MIT Press, 2001.

[9] Konrad Dettner and Werner Peters. *Lehrbuch der Entomologie*. GUSTAV FISCHER, 1999.

[10] James D. Foley, Andries Van Dam, Steven K. Feiner, and John F. Hughes. *Computer Graphics: Principles and Practice*. Addision-Wesley, 1990.

[11] D. Heusser and R. Wehner. The visual centring response in desert ants - *Cataglyphis fortis*. *Experimental Biology*, 205:585–590, 2002.

[12] M. Kohler and R. Wehner. Idiosyncratic route-based memories in desert ants, *Melophorus bagoti*: How do they interact with path-integration vectors? *Neurobiol. Learn. Mem.*, 83:1–12, 2005.

[13] T. Labhart and E.P. Meyer. Neural mechanisms in insect navigation: polarization compass and odometer. *Curr. Opinion in Neurobiology*, 12:707–714, 2002.

[14] D. Lambrinos, R. Möller, T. Labhart, R. Pfeifer, and R. Wehner. A mobile robot employing insect strategies for navigation. *Robotics and Autonomous Systems*, 30:39–64, 2000.

[15] R. Möller. Insects could exploit UV-green contrast for landmark navigation. *Journal of Theoretical Biology*, 214:619–631, 2002.

[16] M. Müller and R. Wehner. Path integration in desert ants - *Cataglyphis fortis*. *Proc. Natl. Acad. Sci.*, 85:5287–5290, 1988.

[17] A. Ohya, A. Kosaka, and A. Kak. Vision-based navigation by a mobile robot with obstacle avoidance using single-camera vision and ultrasonic sensing. *IEEE Transactions on robotics and Automation*, 14:969–978, 1998.

[18] Teng Leng Ooi, Bing Wu, and Zijiang J. He. Distance determined by the angular declination below the horizon. *Nature*, 414:197–200, 2001.

[19] S. C. Pratt, S. E. Brooks, and N. R. Franks. The use of edges in visual navigation by the ant *Leptothorax albipennis*. *Ethology*, 107:1125–1136, 2001.

[20] B Ronacher and R Wehner. Desert ants *Cataglyphis fortis* use self-induced optic flow to measure distances travelled. *J. Comp. Physiol. A*, 177:21–27, 1995.

[21] S. Sommer and R. Wehner. Vector navigation in desert ants, *Cataglyphis fortis*: celestial compass cues are essential for the proper use of distance information. *Naturwissenschaften*, 92:468–471, 2005.

[22] Alan Watt. *3D Computer Graphics*. Number 413-414. Addision-Wesley, 1994.

[23] R. Wehner. Taxonomie, Funktionsmorphologie und Zoogeographie der saharischen wüstenameise *Cataglyphis fortis* (Forel 1902) stat. nov. *Senckenbergiana biol.*, 64:89–132, 1983.

[24] R. Wehner. *Animal Homing*, chapter 3: Arthropods, pages 45–144. ed. F. Papi / Chapman and Hall, 1992.

[25] R. Wehner. Desert ant navigation: how miniature brains solve complex tasks. *J. Comp. Physiol. A*, 189:579–588, 2003.

[26] R. Wehner. *The Neurobiology of Spatial Behaviour*, chapter : Path integration in insects, pages 9–30. ed. K. J. Jeffrey / Chapman and Hall, 2003.

[27] R. Wehner, A.C. Marsh, and S. Wehner. Desert ants on a thermal tightrope. *nature*, 357:586–587, 1992.

[28] R. Wehner and F. Räber. Visual spatial memory in desert ants, *Cataglyphis bicolor* (hymenoptera: Formicidae). *Experientia*, 35:1569–1571, 1979.

[29] R. Wehner and M.V. Srinivasan. Searching behaviour of desert ants, genus *Cataglyphis* (Formicidae, Hymenoptera). *J. Comparative Physiology*, 142:315–338, 1981.

[30] R. Wehner and S. Wehner. Insect navigation: use of maps or ariadne's thread? *Ethol. Ecol. Evol.*, 2:27–48, 1990.

[31] E.O. Wilson. Pheromones. *Scientific American*, 208:100–114, 1963.

[32] S. Wohlgemuth, B. Ronacher, and R. Wehner. Distance estimation in the third dimension in desert ants. *J. Comp. Physiol. A*, 188:273–281, 2002.

[33] C.P. E. Zollikofer, R. Wehner, and T. Fukushi. Optical scaling in conspecific cataglyphis ants. *The Journal of Experimental Biology*, 198:1637–1646, 1995.