Lab Report:
Investigating Jumping Spider Vision

Aleke Nolte
Supervisors: Dr. Tom Gheysens
Dr. Daniel Hennes
Prof. Dr. sc. nat. Verena V. Hafner

February 9, 2015

Abstract

Jumping spiders are capable of estimating the distance to their prey relying only on the information from one of their main eyes. Recently it has been suggested that jumping spiders perform this estimation based on image defocus cues. In oder to gain insight into the mechanisms involved in this blur-to-distance mapping as performed by the spider and to judge whether inspirations can be drawn from spider vision for Depth-from-Defocus computer vision algorithms, we model the relevant spider eye with the 3D software Blender and render scenes how the spider would see them. We test a simple and well known Depth-from-Defocus algorithm on this dataset and show why it is not capable of estimating distances correctly. We conclude that our model is too abstract to allow inferences to be drawn about the spider's Depth-from-Defocus mechanism. Further we propose another straightforward way to quantify defocus and suggest additional aspects to be included in the model.

1 Introduction

Computer Vision (CV) algorithms that deduce distances from two or more defocussed images of a scene, so called Depth-from-Defocus (DFD) algorithms, have first been introduced in the late 1980s and the beginning of the 1990s (e.g [1, 2]) and have been build upon subsequently to increase performance and robustness (e.g. [3]).
Recently, Nagata et al. [4] presented a study which provides strong evidence that jumping spiders also judge the distance to their prey based on defocus cues. In the first part of this study, all but one of the spider's front facing

anterior median (AM) eyes were occluded. The spider was then presented a fruit fly at different distances and jumping accuracy during hunting was accessed. It was found that there is no significant difference in performance to an unblinded spider, indicating that the spider relies on monocular cues from the AM eyes for distance estimation. What is special about the AM eye is that the photoreceptor cells are distributed over four layers, of which the two bottommost layers, Layer 1 (L1) and Layer 2 (L2) yield the input for the DFD estimation: images projected onto the retina are blurry on L2 and sharp on L1 (Figure 2). Using the achromatic aberrations of the eye's lens in the second part of their study (more details in Section 1.1), Nagata et al show that the amount of blur on these layers indeed influences the jumping distance.

The question now arises if both CV-DFD algorithms and the spider DFD mechanism are based on the same high level principles and if not, if CV-DFD algorithms can be improved by learning from the spider's DFD mechanism. Particularly interesting is that the spider achieves an apparently very accurate depth estimation with very simple components: A single lens and very few photoreceptors (e.g. an AM eye of the spider species *Metaphidippus aeneolus* only has about 1200 photoreceptors in total [5]).

A DFD sensor build of such basic components would be of great interest for use in space: If only basic components are needed the sensor is likely to be low in cost. Low cost equipment is highly suitable for simple spacecrafts like cubesats, and an optical distance sensor would be advantageous for formation flying and other types of swarm behaviour. A low cost optical distance sensor would also be of use in debris removal, during docking and landing and as a general fall-back or assisting option for other distance sensing equipment such as laser detection and ranging systems (LADAR), sound navigation and ranging systems (SONAR) and stereo vision systems. Furthermore a system consisting essentially of only lenses and sensors and no mechanical parts would be less fault-prone and lower in power consumption than above mentioned methods.

In this work we take the first steps in investigating if spider vision has the potential to advance such a vision sensor. To this end, we create a model of the anterior median (AM) eye of spider species *Metaphiddipus aeneolus* to see what kind of images are created on the spider retina. Simply put, we want to show "what the spider sees". We then test a well known but basic DFD-algorithm [2] on these images to see if the spider's depth assessment performance can be explained by the principles of this algorithm.

This paper is structured as follows: In the next Section 1.1 we describe the morphology of the AM eye of *M. aeneolus* as found by Land [5] and present the study above mentioned by Nagata et al. in more detail. In Section 2 we

specify the abstractions made and the modeling choices for our model of the spider eye in the 3D graphics software Blender [6]. Section 4 describes the DFD algorithm used and Section 5 delineates the datasets constructed and the experiments performed, as well as an investigation on why the algorithm fails to perform on our spider datasets. The paper closes with a discussion on further elements to be included in the model and a simple possible alternative to the algorithm used.

1.1 The anterior median spider eye

The photoreceptors of the retina of the AM eyes of the jumping spider are distributed over four layers (see Figure 3). According to an analysis by Land from 1969 [5] the two topmost layers always receive defocussed images, as the corresponding conjugate object planes lie behind, and not in front of the spider's head. Recently, the two bottommost layers have been found to be essential for the judgement of distances: Nagata et al. found that these layers contain mostly green light sensitive photo receptors and have shown in an experiment that a jumping spider performs accurate jumps when viewing targets in green, but not in red light.
Furthermore, they showed that the spiders consistently jumped to short in red light, thus underestimating the distance to their prey. Nagata et al. found that due to achromatic aberrations of the spider eye's lens an object viewed in red light results in the same amount of blur on the second retina layer (L2) as the same object at a closer distance when viewed in green light (see Figure 2). Thus the underestimation of the distance to the prey in red light is strong evidence that the spider judges the distance on defocus cues.

Morphology The anterior median spider eye resembles a long tube (Figure 1), with the very curved cornea on one end and a boomerang shaped four layer retina on the other end (Figure 4). The four retina layers extend over a range of ca $50\mu m$ with the distances between layers and layer thicknesses as indicated in Figure 3. On the retina, the receptors are arranged in a hexagonal lattice, with denser spacing closer to the optical axis and a more coarse spacing towards the periphery. The spacing in L2 is overall more coarse than in L1, the minimum receptor spacing found in L1 is $1.7\mu m$.

2 Modeling the Spider Eye

We model the AM eye of *M. aeneolus* with the help of the 3D graphics software *Blender* [6]. To achieve physically accurate results of how light

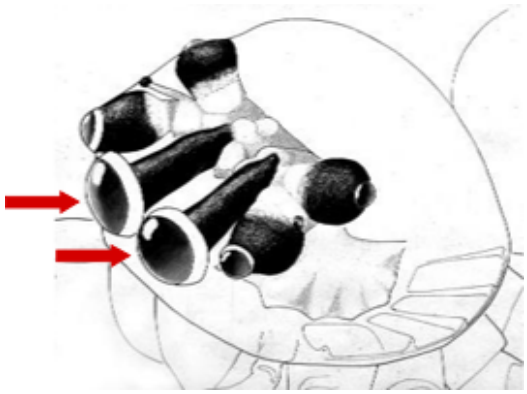


Figure 1: Example shapes of jumping spider eyes. The anterior median eyes (red arrows) resemble a long tube with a very curved cornea in the front. `http://tolweb.org/accessory/Jumping_Spider_Vision?acc_id=1946`, retrieved (12/15/2014)

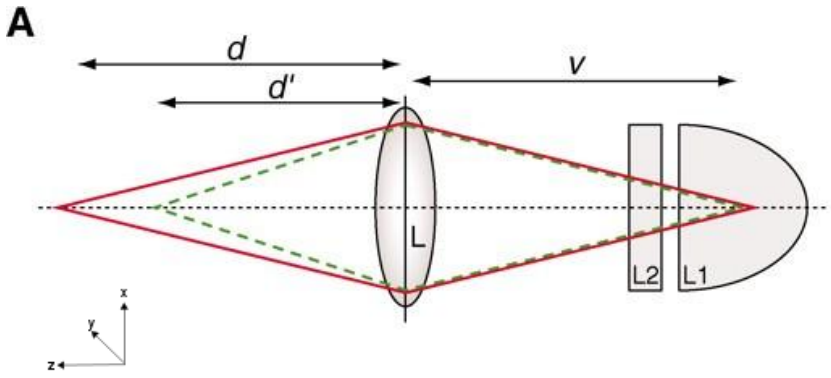


Figure 2: Objects at distances d in red light and d' in green light result in the same amount of blur on Layer 2. (Figure taken from [4].)

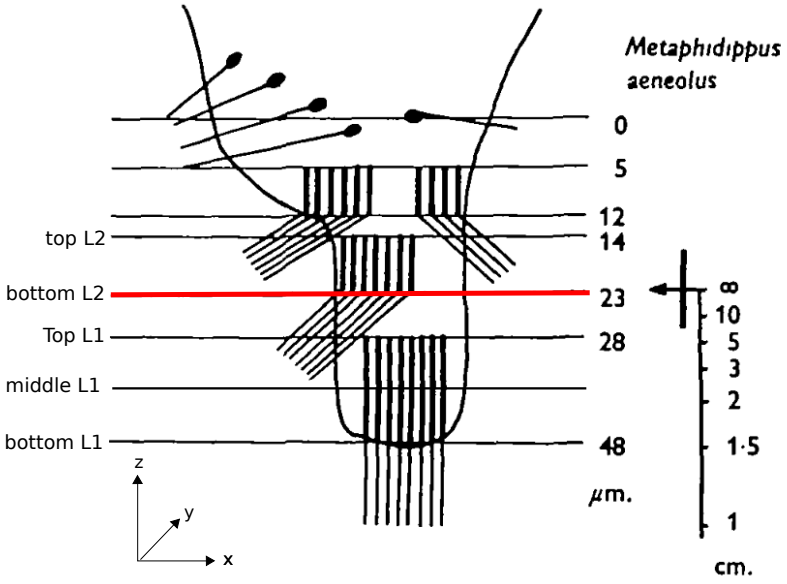


Figure 3: Arrangement of photoreceptor layers of *M. aeneolus*'s retina. Light from the lens enters the figure from the top. The bar on the right indicates the object distances that are conjugate to horizontal planes on the retina. The red line highlights the location of the focal plane. The legend on the left indicates the location of the image planes used in our dataset. (Figure slightly modified from [5] to improve clarity).

is refracted through lens and posterior chamber we choose *LuxRender*[1], a physically based rendering engine for the ray tracing part in place of Blender's build-in *Cycles* engine. *Cycles* does not model glass well and is thus less suitable for an optical model, while *LuxRender* traces light according to mathematical models based on physical phenomena and is thus more suitable. In oder to facilitate the handling of the model, all used values are scaled by a factor of 10000.

Modeling lens and posterior chamber In our model the eye is represented by a thick lens ($r_1 = 217\mu m$, $r_2 = -525\mu m$, $d = 236\mu m$) with refractive index $n = 1.41$[2], enclosed by a black tube [3].
The posterior chamber of the eye is modeled by setting the refractive index of the back of the lens to that of spider ringer, i.e., $n = 1.335$[4]. Aperture and specific shape of the lens (see Figure 6) are achieved by creating a black

[1]http://www.luxrender.net, retrieved 06 January 2015
[2]LuxRender *glass2* volume with corresponding refractive index
[3]LuxRender *matte* material
[4]LuxRender *glass2* volume with corresponding refractive index

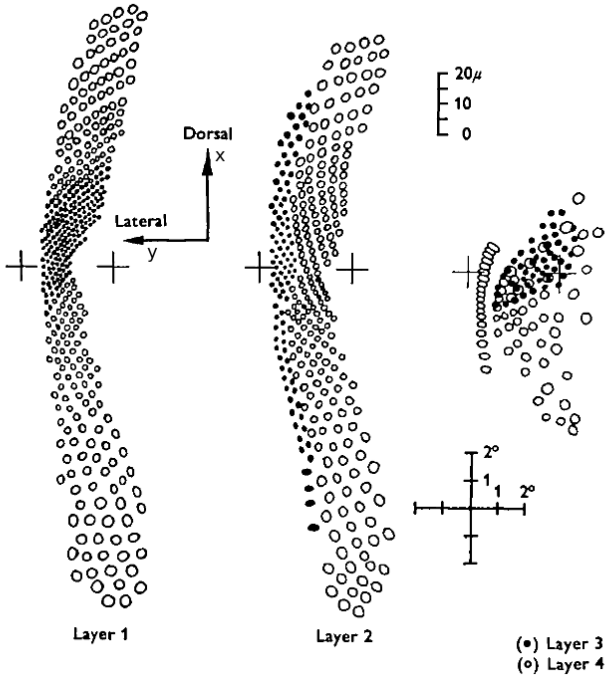


Figure 4: Boomerang shaped receptor grids of Layer 1 (left) and Layer 2 (middle). The rightmost subfigure shows the receptors for Layer 3 and Layer 4 superimposed. The crosses indicate the location where the layers would stack onto each other. For both Layer 1 and Layer 2 receptors are spaced more densely along the optical axis and more coarsely in the periphery. The minimal receptor spacing in Layer 1 is $1.7\mu m$. (Figure taken from [5]).

torus[5] with diameter $d = 200\mu m$ at narrowest point. The resulting model is shown in Figure 5. All the measurement values mentioned above are the same as provided in [5] and summarised in Table 1.

Modeling receptor layers / sensor spacing For simplicity, we do not model the receptor layers as volumes (Figure 3), but simply as 2-dimensional sensor planes. A sensor plane is realized in Blender by creating a plane of *translucent* material (to act as a "film") and placing an orthogonal scene camera behind it to record the image on the film. We record images at locations of the sensor plane in z-axis corresponding to: The top of L2, the focal plane (which coincides with the bottom of L2), the top of L1, the middle of L1 and the bottom of L1. The corresponding back focal distances (BFD) are $450\mu m$, $459\mu m$, $464\mu m$, $474\mu m$ and $485\mu m$, respectively. In our setup, we choose a quadratic film size and base the number of receptors (=

[5]LuxRender *matte* material

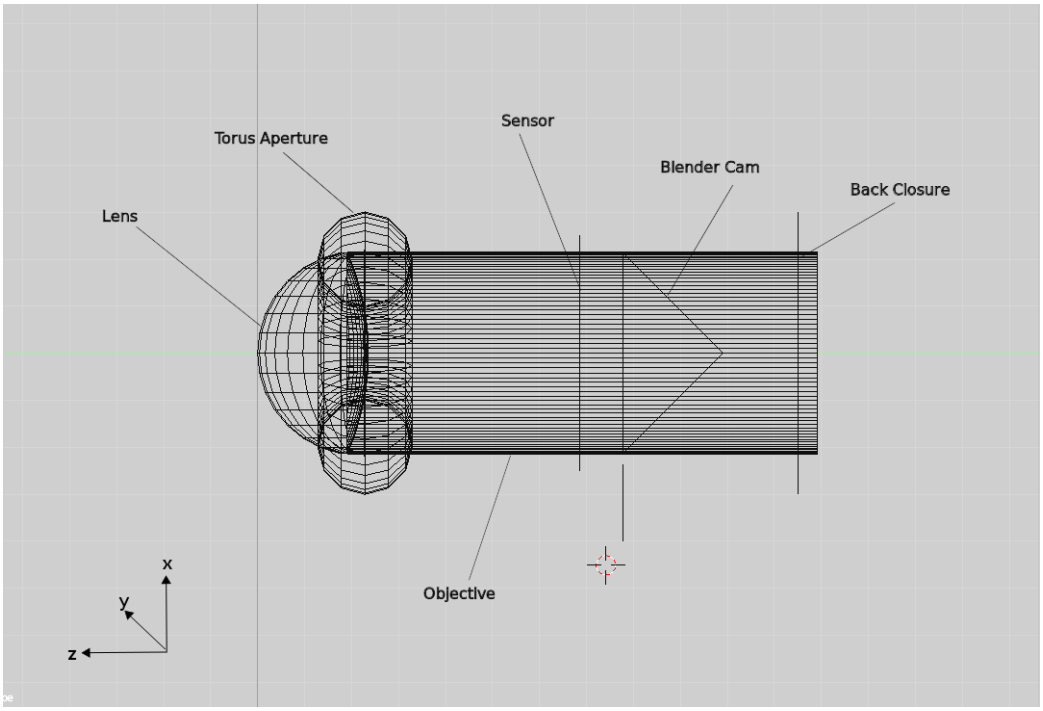


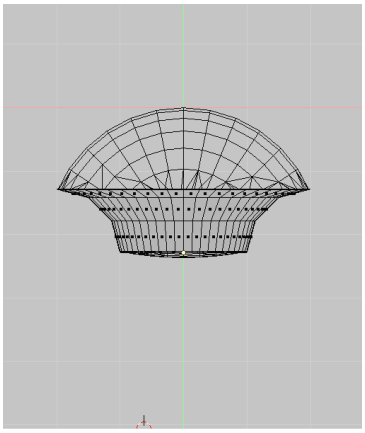
Figure 5: Blender model of the AM eye of *M. aeneolus*. The lens is a LuxRender *glass2* volume with refractive index $n = 1.41$, the posterior chamber is a LuxRender *glass2* volume with refractive index $n = 1.335$. The sensor is of LuxRender *translucent* material. All other materials are of type LuxRender *matte* and of black color to absorb light.

Eye Parameters

r_1	$-r_2$	d	n_{lens}	$n_{\text{postr chamber}}$	f
$217\mu m$	$525\mu m$	$236\mu m$	1.41	1.355	$504\mu m$

Table 1: Parameters of the AM eye of *M. aeneolus*.

pixels) on the closest spacing in L1, resulting in a $117px \times 177px$ film of size $200\mu m \times 200\mu m$ to approximate the receptors of a layer. We use this film to record images for L1 as well as for L2. Even though the structure of the actual receptor layers is boomerang shaped and only measures a few micrometers at its most narrow point, we assume that the spider can emulate a larger retina by moving the retina in x-y direction, assumingly "stitching" the partial images together to form a larger image. This behaviour might also result in higher visual accuracy, emulating a more dense receptor spacing. We thus appropriately assume to model the retina as a square. To account for the possible higher accuracy, we also create a parallel dataset with 370px$\times$370px occupying the $200\mu m \times 200\mu m$ sized film.



Sideview of lens in Blender model

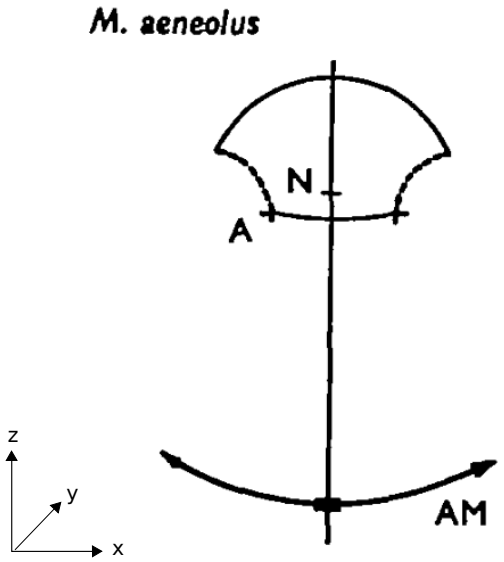


Figure 6: Comparison of AM eye lens shape of *M. aeneolus* in the Blender model (left) with the lens shape as reported by Land (right, image taken from [5])

3 Dataset

We generate the spider dataset by rendering four different images ("objects") as viewed through the spider eye, two artificial and two "natural" ones (see Figure 7). The artificial ones are a checkerboard texture, where the size of one square corresponds to the size of a fruit fly (approx. 5 mm) and a black circle in front of a neutral background. Here, the diameter of the circle also corresponded to 5mm. The artificial textures where chosen like this to a) be able to visually judge the defocus around edges and b) get an impression how the size of prey on the sensor changes for different distances from spider to prey. These textures however have a low frequency content and lack detail. The DFD algorithm is reported to not perform well with low-frequency content images [2]. To account for this we include two additional natural images. It should be noted that even though these images display a jumping spider and a fruit fly, these are not to scale, but much larger than their real counterparts. Considering the possibility that the effective resolution of the spider eye is higher due to eye movement, we also create a set of higher resolution ($370px \times 370px$) renderings for the natural images. We render images for each object placed at distances $D = \{1.5, 3 \text{ and } 6\text{cm}\}$ from the lens. These distances approximately correspond to the conjugate planes (in object space) of the Bottom of L1, the middle of L1 and the top of L1, respectively and are chosen so to ensure that "good" images , i.e.

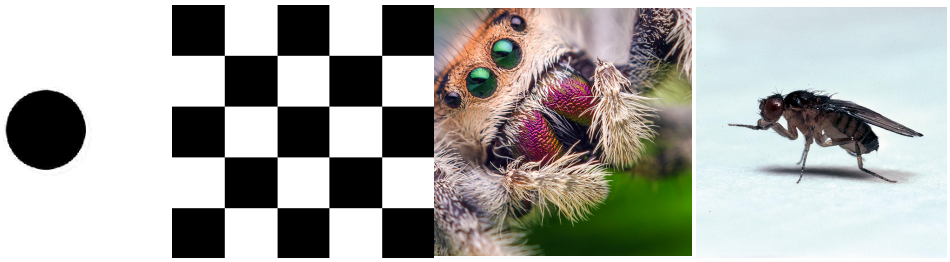


Figure 7: Images used to generate the spider dataset. These are the "objects" that are placed in front of the lens to be viewed *through* the lens. (Spider image from `http://commons.wikimedia.org/wiki/File:Female_Jumping_Spider_-_Phidippus_regius_-_Florida.jpg`, retrieved 04/01/2014; fruit fly image from `http://www.carolina.com/drosophila-fruit-fly-genetics/drosophila-living-ebony-chromosome-3-mutant/172500.pr`, retrieved 14/01/2014.

images with least amount of blur, are part of the dataset. Examples from the datasets are show in Figure 8.

4 Depth from Defocus Algorithm

In the following we will present the reasoning underlying DFD algorithms and describe a basic algorithm as proposed by Subbarao in 1988 [2]. Even though the algorithm has been improved upon in many ways since it was first proposed (e.g. [3]), most improvements address the *image overlap problem* - the problem that when segmenting an image into patches to estimate the distance of each patch, each patch is influenced by objects in neighboring patches due to the spread of the defocus. In our setup however, the objects we consider are planes perpendicular to the optical axis so that the distances are constant over the whole image. Accordingly these improvements are not expected to increase the performance in our scenario, so that we only use the original algorithm.

4.1 Subbarao's DFD algorithm

If an object is not in focus, the amount of blur in the image can provide information about the distance of the object. Given that we know the camera's parameters (focal length f, aperture A and the lens to sensor distance v) we can calculate the distance by basic geometry and Gauss' lens formula. Gauss equation assumes a thin lens and relates the object distance and the

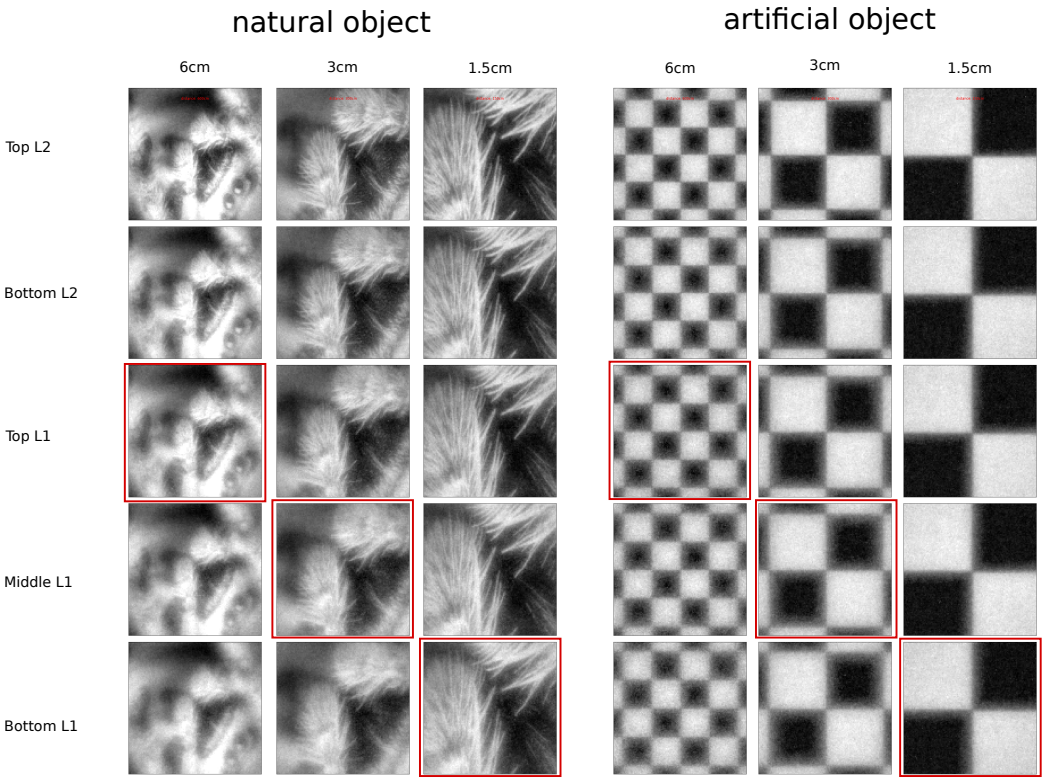


Figure 8: Examples for one natural and one artificial texture dataset. Shown are images at the sensor planes corresponding to Top of L2, bottom of L2, top of L1, middle of L1 and bottom of L1 (top to bottom) for object distances corresponding to 6cm, 3cm and 1.5cm, (left to right). The red squares indicate the images that should be least blurry according to Gauss' lens equation (Eq. 1).

focal length of the lens to the distance of the focussed image:

$$\frac{1}{f} = \frac{1}{D} + \frac{1}{v}\,, \tag{1}$$

where D is the distance of the object to the lens.
The diameter d of the blur circle is related to the other camera parameters

$$d = A\,v\,(\frac{1}{f} - \frac{1}{D} - \frac{1}{v})\,, \tag{2}$$

and the actual observed blur circle radius σ then depends on the camera constant ρ (which depends in parts on the pixel resolution and in part on other camera properties)

$$\sigma = \frac{\rho}{2} d\,. \tag{3}$$

If the diameter of the blur circle is known Eq. (2) can easily be solved for object distance D. Accordingly, the basis of most DFD algorithms including

Subbaraos's algorithm, is the estimation of blur from two or more defocussed images.
The basic premise of the algorithm is that an out of focus image can be created from a sharp image by convolving the sharp image with a point spread function (PSF) that corresponds to the blur. For simplicity the PSF is often assumed to be a two dimensional Gaussian

$$h(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4}$$

with spread parameter σ. A blurred image $g(x,y)$ can thus be obtained from a sharp image $f(x,y)$ by convolving the sharp image with the PSF

$$g(x,y) = h(x,y) * f(x,y)\,. \tag{5}$$

Convolution in the spatial domain corresponds to multiplication in the frequency domain. Thus, when considering the blurry images in the frequency domain we can eliminate the need for a sharp image:

$$\begin{aligned} G_k(\omega,\nu) &= H_k(\omega,\nu)F_k(\omega,\nu) \\ \frac{G_1(\omega,\nu)}{G_2(\omega,\nu)} &= \frac{H_1(\omega,\nu)}{H_2(\omega,\nu)}\,. \end{aligned}$$

Using the frequency space representation of Eq. (4) this leads to

$$\frac{H_1(\omega,\nu)}{H_2(\omega,\nu)} = \exp\left(-\frac{1}{2}(\omega^2+\nu^2)(\sigma_1^2-\sigma_2^2)\right)\,. \tag{6}$$

Considering the power density spectra $P(\omega,\nu)$ of the transform and rearranging allows to extract the relative defocus

$$\sigma_1^2 - \sigma_2^2 = \frac{-2}{\omega^2+\nu^2} log \frac{P_1(\omega,\nu)}{P_2(\omega,\nu)}\,. \tag{7}$$

In order to obtain a more robust estimation, the relative defocus is averaged over a region in frequency space

$$C = \frac{1}{N}\sum_{\omega}\sum_{\nu} \frac{-2}{\omega^2+\nu^2} log \frac{P_1(\omega,\nu)}{p_2(\omega,\nu)}\,, \tag{8}$$

where $P_1(\omega,\nu) \neq P_2(\omega,\nu)$ and N the number of frequency samples.
It is then possible to solve for the blur of one of the images, e.g. σ_2, by solving the following quadratic equation:

$$(\alpha^2-1)\sigma_2^2 + 2\alpha\beta\sigma_2 + \beta^2 = C \tag{9}$$

with

$$\alpha = \frac{v_1}{v_2} \tag{10}$$

and

$$\beta = \rho v_1 \frac{A}{2}(\frac{1}{v_2} - \frac{1}{v_1})\,. \tag{11}$$

Using the obtained σ_2 in Eq. 3 and Eq. 2 allows to solve for the distance D.

5 Experiments

5.1 Pre-experiment on *kixor* dataset

To test the functionality of the algorithm we first tested it on a small dataset[6] of defocussed images obtained with an off the shelf camera. The dataset does not contain the exact distances of the objects pictured, nor does it include the exact lens to sensor distances but instead it reports the distance the camera focusses on. Using Gauss' lens formula, and by visually determining which part of the image is sharp, both u and D can be approximated. Testing the algorithm on this dataset yields results in the correct order of magnitude if the blur in the images is not too high. The exact results also depend on the frequencies used to calculate C. Examples are shown in Figure 9.

5.2 Experiments on spider dataset

Testing the algorithm on the spider data did not give consistent results. For most object- and sensor distances, the estimated distance was neither in the right order of magnitude, nor did it reflect the ordering of distances (giving higher estimates to objects further away and lower estimates to objects that were closer).

Figure 10 shows the estimates given by the algorithm for different object distances and for the different spider datasets.

In the following we analyse why the algorithm performs poorly on the spider datasets.

5.2.1 Misleading frequency content

As described in Section 4.1 the estimation of the blur of the images is based on the "difference" of the two images in the frequency domain. Accordingly, if there is not enough or misleading frequency information in the image, the algorithm is not expected to work. To address this we test the algorithm

[6] `http://www.kixor.net/school/2008spring/comp776/project/results/`, retrieved 16/12/2014

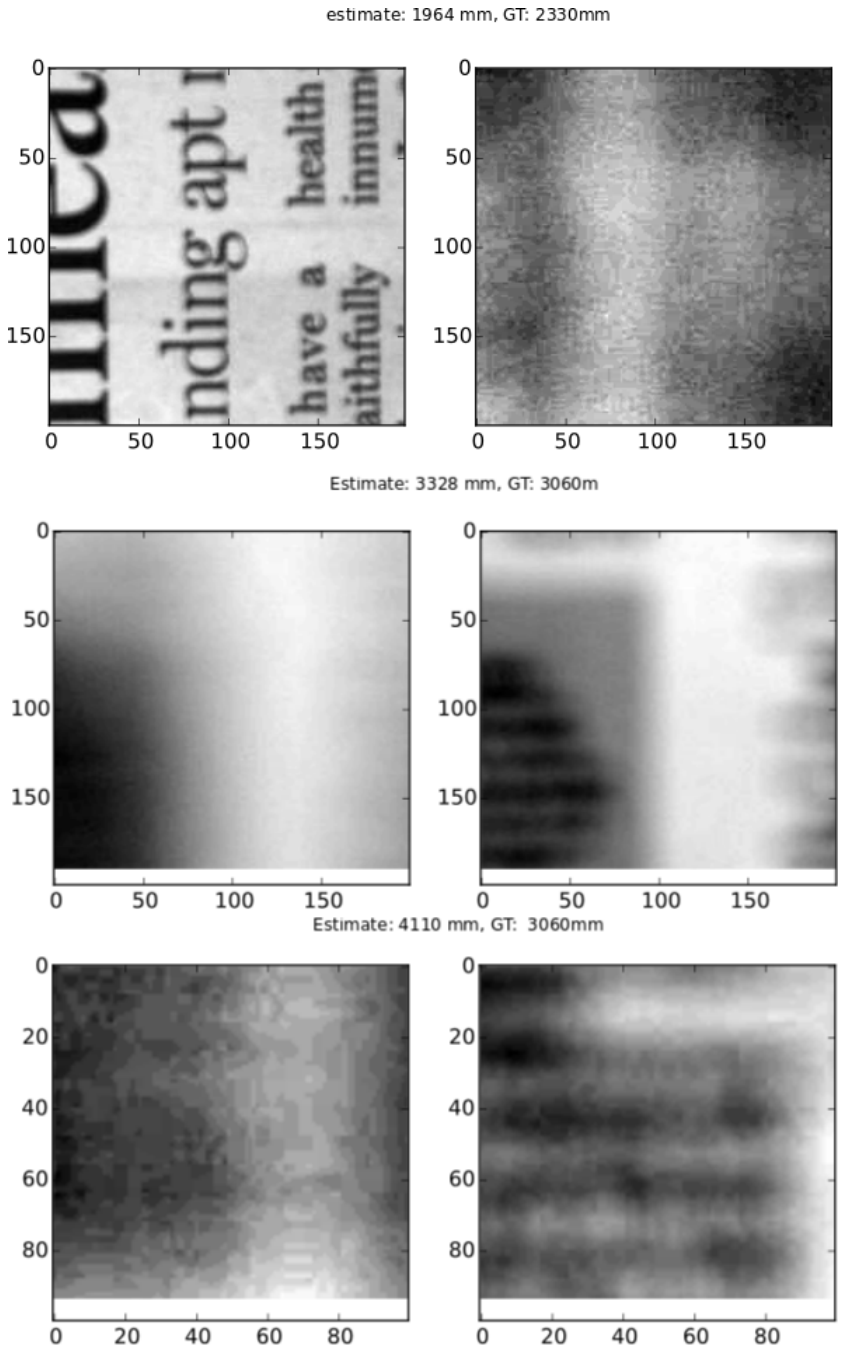


Figure 9: Image patches extracted from *kixor* dataset. Shown are three examples of defocussed image pairs and the estimated as well as the actual distances of the objects.

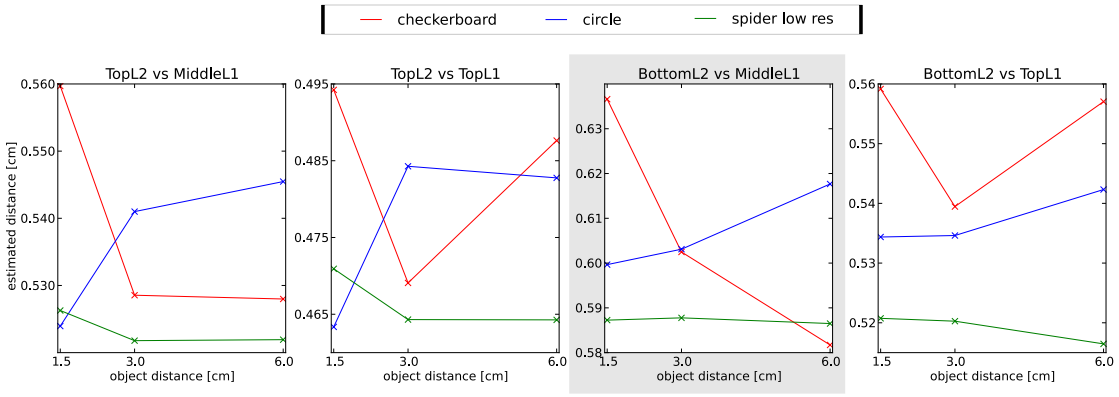


Figure 10: Depth estimates for the low resolution (117px × 117px) spider dataset. Results are shown for a subset of pairings from image planes in L2 and L1. The highlighted subfigure is the pairing that is assumed in spiders.

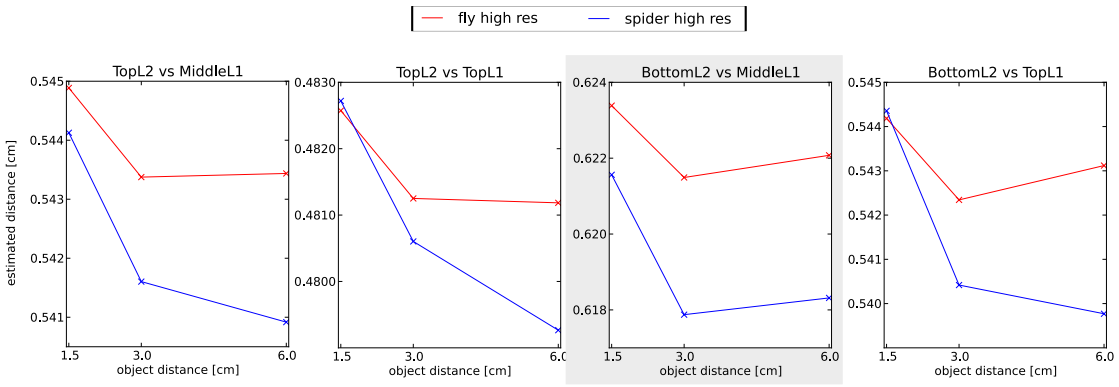


Figure 11: Depth estimates for the high resolution (370px × 370px) spider dataset. Results are shown for a subset of pairings from image planes in L2 and L1. The highlighted subfigure is the pairing that is assumed in spiders.

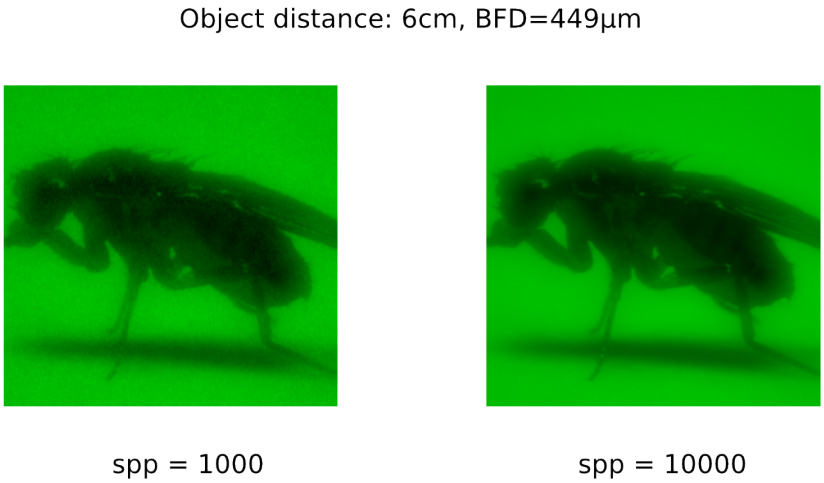


Figure 12: Amount of noise for different rendering times. Left: 1000 samples per pixel, right: 10000 samples per pixel. Images are taken from the high resolution dataset 370px × 370px)

on a dataset with low frequency and normal frequency content (checkers, circle and "natural image" dataset). The performance on the natural image dataset is qualitatively the same as on the artificial datasets, indicating that the low frequency content of the image is not the reason for the failure.
Also, the receptor grid is quite coarse (117px × 117px) so that fine details which could provide valuable blurred edges (and with that more frequency information) might not be captured. However, the algorithm's performance does not increase on the high resolution natural images dataset (370px × 370px) (Figure 11).

Image noise levels The test datasets contain some noise due to the rendering process. During rendering, light paths are traced from Blender's light source to the viewed object, through the lens, and onto the film. Each found path is called "a sample". The process of rendering is computationally costly and thus the numbers of samples per pixel (spp) is limited to 1000 in our original dataset. As shown in the left of Figure 12, the resulting image is slightly noisy. The noise is the same for different object distances, and thus the frequencies corresponding to the noise are the same for both images, which in turn can influence the blur estimate.
We test the algorithm on datasets with lower amounts of noise: on low pass filtered versions of the original dataset and a dataset with $spp = 10000$. (Figure 12 right). The results of the algorithm remain qualitatively the same as shown in Figures 10 and 11. Accordingly, neither the rendering noise nor

the resolution of the images are the reason for the poor performance of the algorithm.

5.2.2 Focal lengths

The focal length for *M. aeneolus* as measured by Land is $512\mu m$, while the calculated focal length for a thin lens in air and water amounts to $504\mu m$. Using the above (Section 2) lens parameters (refractive indices, radii of curvature and thickness of the lens) allows to calculate the back focal length of the lens (BFL), that is the distance between the back of the lens and the focal point, which amounts to $459\mu m$. However, these calculations do not take spherical aberrations into account. Due to spherical aberrations an image formed by a very curved lens may still be blurry, even though according to Gauss' lens formula it should be in focus. The sensor distance for "best" focus is then the distance at which the "circle of confusion" is smallest.
When setting up the sublayers for L1 and L2 relative to $BFL = 459\mu m$, the expected trajectory from defocussed to focussed and back to defocused could not be observed in our renderings (Figure 8).

Taking into account that Land's description of the eye may not be complete and to investigate a case that is closer to the considerations presented by Nagata [4] (more clearly focussed and defocussed images) we create an additional set of test images with sensor layer distances relative to the "best" focus, namely the *BFL400*-dataset. We determined the best focus with the help of the autofocus function of the commercial optical software *Zemax*[7]. Unlike in the ideal lens calculations Zemax's autofocus function calculates the back focal length based on the smallest circle of confusion, resulting in a BFL of $400\mu m$.
For the layers placed relative to this value, a different blur profile can be observed (Figure 13). However, testing the algorithm on this dataset results in qualitatively the same performance as the mentioned experiments. This indicates that the poor performance of the algorithm is not due to an incorrect value for the focal length.

5.2.3 Reverse calculations: Which values of σ and C would we expect?

In order to further investigate why the algorithm performs poorly on the spider dataset, we calculate the values for C and σ which the algorithm would expect in order to yield the correct results. Figure 14 shows a comparison of the calculated values for C and the values for C obtained by using the images from the spider dataset in Eq 8, $f = 459\mu m$ and $D \in \{1.5\text{cm}, 3\text{cm}, 6\text{cm}\}$. It

[7] https://www.zemax.com

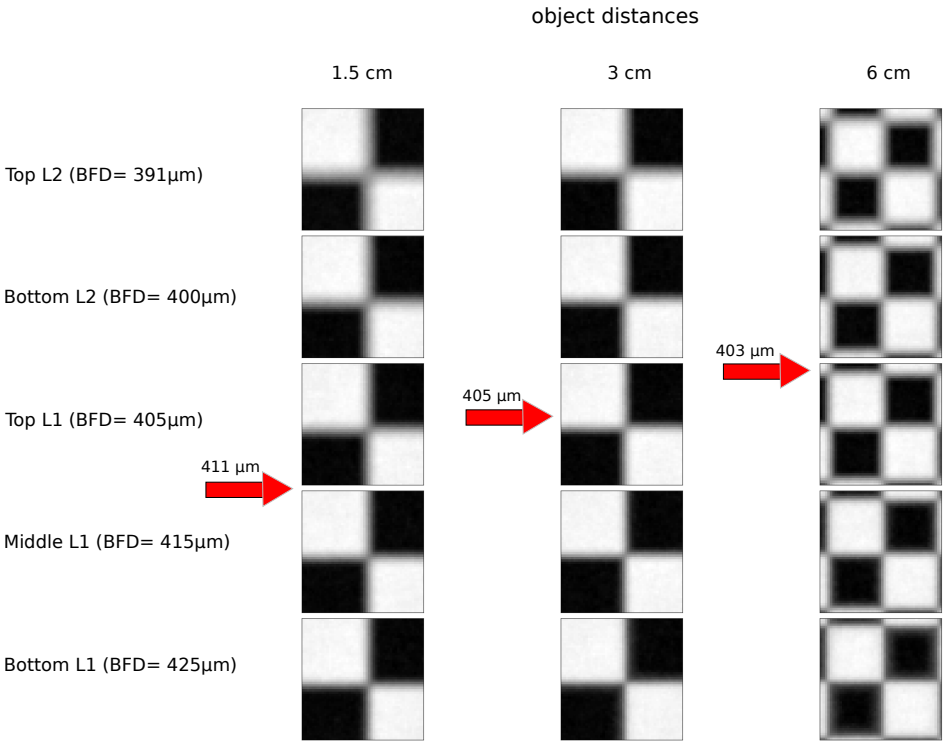


Figure 13: Layers placed such that the bottom of L2 coincides with $BFD = 400\mu m$. Red arrows indicate at which back focal distance (BFD) the image should be sharpest assuming a thin lens and a focal length of $f = 400\mu m$ (calculated with Gauss' lens formula).

can be seen that obtained results do not agree nor follow the general trend of the calculated results.

To gain insight why the C-values do not yield sensible results even for the "good-natured" high-frequency content and low-noise images, we generate artificial blurry images and compare them with the rendered spider dataset images (Figure 15). The images are generated by simply convolving the initial sharp image with a Gaussian filter with σ corresponding to an object distance of $D = 6cm$ and sensor plane distances of $v \in \{434\mu m, 444\mu m, 459\mu m, 464\mu m, 474\mu m\}$ as described in Eq 5. The size of the blur circle is computed in microns and then translated into pixels by setting camera parameter $k = \frac{117\text{px}}{200\mu m}$. Here the last three sensor distances correspond to the bottom of L2, top of L1 and middle of L1. The first two sensor distances are included to further illustrate the difference between the blur profiles for the rendered and convolved images.
Comparing the rendered with the convolved images shows two important

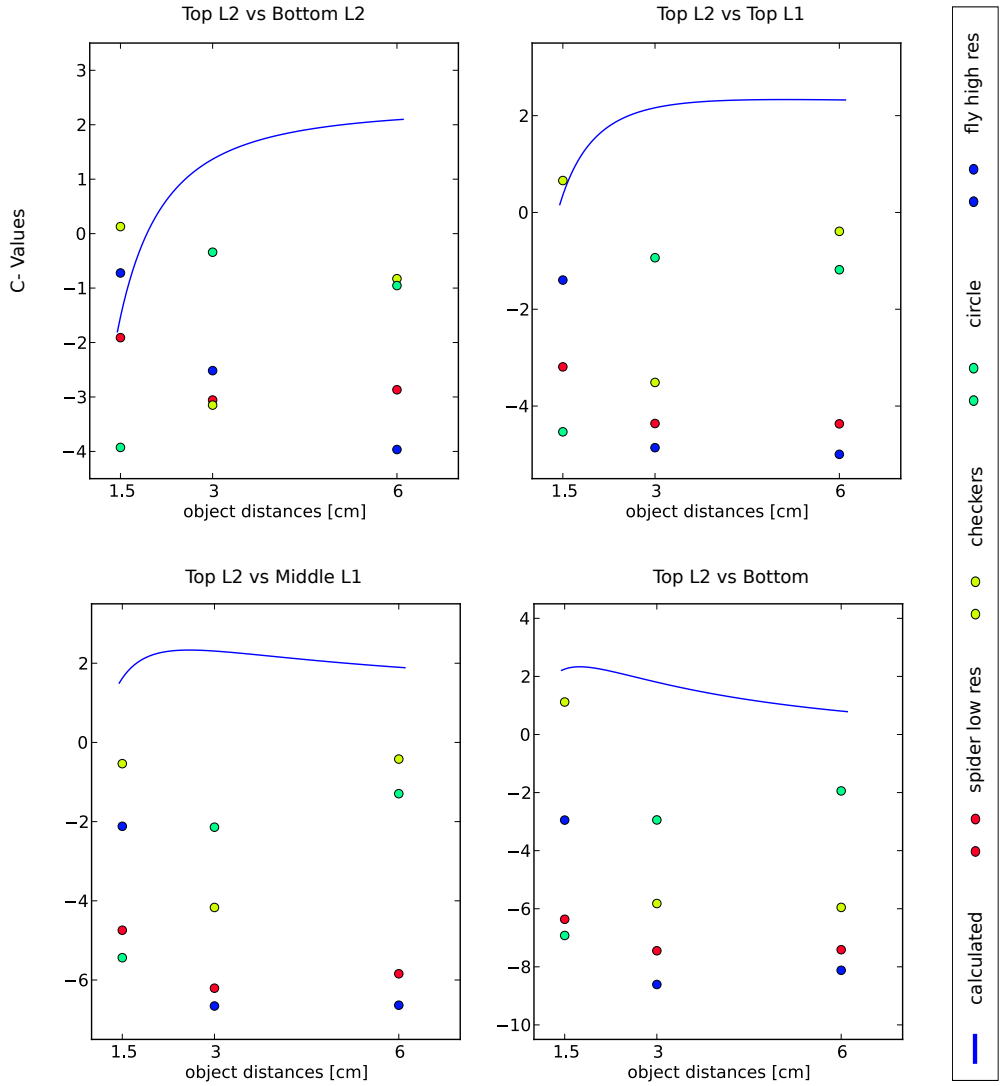


Figure 14: Comparing calculated values of C with the values for C obtained by using Eq 8 with the spider datasets. The titles of the subfigures indicate the receptor layers that "recorded" the images used for obtaining C. Here, we fix the top of L2 as one of the images.

differences: Firstly, the rendered images are much more distorted due to the very curved lens. This also results in a blur amount that varies over image location: the image is sharper in the center and more blurry towards the periphery. Secondly, the rendered images appear much more blurred than the convolved images. Also, the rendered image is at its best at sensor planes closer to the lens and gets worse on sensor planes that are further away, while the convolved image is best at $BFL = 464\mu m$ and worse at distances closer to and further away from the lens.
A conclusion that can be drawn from this is that the thick lens of the spider eye has too many aberrations for the image forming process to be sensibly approximated by convolution with a Gaussian PSF, and that the spider must thus be using a different mechanism.

6 Discussion and Conclusion

Algorithm As shown in the previous section, Subbarao's simple DFD algorithm fails to provide sensible distance estimates on the spider dataset. The reason for this is that the algorithm assumes an ideal thin lens, while the lens of the spider eye is a thick lens with a large amount of uncorrected aberrations. Thus the PSF is not Gaussian and thus the basic premise that a defocussed image can be modeled as the corresponding sharp image convolved with a Gaussian filter is not fulfilled.
In principle, it would be possible to adjust the algorithm to other PSFs at the cost of losing the simple closed form solution [2]. However, due to the aberrations of the lens, the PSFs differ for different sensor and object distances, which would result in many complications of the algorithm.
Alternatively, one could also think of a more straightforward spatial domain algorithm: Given that the spider has already identified the prey in its field of view, comparing the numbers of active receptors between the layers and the difference in light intensity for adjacent receptors might already suffice as a decent distance estimate.
With this method in mind it might also be sensible to look into neuronal inspired algorithms, e.g. how DFD-functionality might be achieved with a spiking neural network.

Eye model and dataset Apart from possible adjustments to Subbarao's algorithm, there are points for discussion on what the spider actually sees and how a distance measurement may be generated from this. For one, the images as projected onto the retina layers are much more blurry than expected. Particularly, when considering Figure 8 it is hard to determine which of the images is the least blurry image for a particular object distance. Even for the

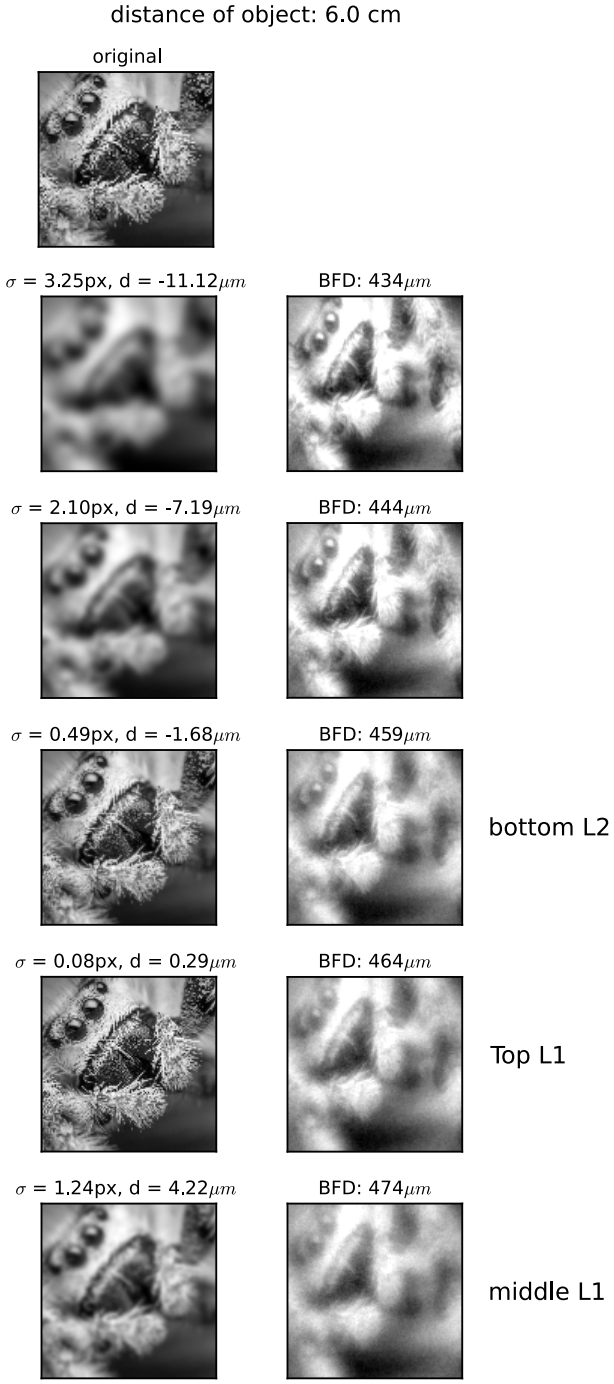


Figure 15: Comparing rendered images (right) with images generated by convolving the original image with Gaussian spread function (left). Spread is parametrized by σ; values for d are obtained by Equation 2. K is $\frac{117px}{200\mu m}$.

case of the shifted receptor layers (bottom of L2 at $BFD = 400\mu m$) which is not biologically accurate but results in a blur profile that is closer to the expected one (see Figure 13 for shifted receptor layer images and left of Figure15 for expected profile), it is still hard to judge where exactly the image is sharpest. If the best image is not clearly distinguishable, and projected images are very similar over a range of sensor distances, information is lost and thus accurate determination of the object's distance will be no longer possible.

Before concluding that the spider accomplishes the impossible task of estimating distances from images that do contain sufficient information, it makes sense to consider what the spider actually sees. It is important to point out that the images generated with our Blender model reflect the light which would fall onto un-occluded planes behind the lens. However, the receptors in spider eyes are not 2D planes, but volumes. Accordingly, the thickness of the receptor layers may play a further role. It seems likely that photons can react with the receptive segments of the photoreceptor over its whole length (i.e reactive segment is a long as the layer is thick), so that the "image" as received by the receptors may be rather an integration of all image planes than the single images used in our dataset. Additionally, in order to reach e.g the "bottom of L1" a photon must first pass all of L2 and the top parts of the receptive segments of L1 without being absorbed. So in order to model accurately how much light each receptor receives, it would be necessary to have an idea on how likely photons are to be absorbed on different locations of the retina.

In this volume-scenario either the information on where exactly on the layer the image is sharpest or how the defocus varies over the length of the layer is lost, unless there are some unknown intracellular mechanisms preserving it.

Future work From the performance of the algorithm and above considerations we can conclude that the model as based on Land's findings [5] is too abstract to be able to make conclusions about the spider's DFD mechanism. The present understanding of the components of the jumping spider eye is still incomplete, but further insights may be gained by including more of the known details in the model.

A possible next step would be to model the receptor spacing (Figure 4) more accurately. The particular spacing and layout of the receptors may in part compensate for aberrations. An additional step may be to include more details on the vertical arrangement of the receptors and to move from a static model to a model which accounts for retina movements: According to Blest et al. [7] the receptor ends of both L2 and L1 form a stair-like structure, which might provide depth information when the eye performs scanning movements.

Also, in reality L1 and L2 are not parallel to each other but oriented in a angle. This arrangement could also provide additional depth information when comparing inputs to the layers. These findings point to very different mechanisms for determining distances than the ones used in CV DFD algorithms.
Based on measurements in jumping spider species *Plexippus* Blest also suggest that the interface between the posterior chamber of the eye and the receptors acts as a second lens. This would change the optical system and thus the way light falls onto the receptors, potentially resulting in better distinguishable depth-profiles.

Conclusions We present images created by a model of the AM eye of spider species *M. aeneolus* and analyse why Subbarao's DFD algorithm fails to estimate distance accurately on these images. We conclude that the model as based on considerations and measurements by Land [5]is too abstact to allow inferences to be drawn about the spider's DFD mechanism. We propose another straightforward way to quantify defocus and suggest further aspects to be included into the model.

Acknowledgements

Many thanks to Charlotte Pachot for help with Zemax and expertise in the domain of experimental and theoretical optics and to Michael Klemm for feedback on Blender and Luxrender aspects of the eye model. I would also like to thank Thijs Versloot, Isabelle Dicaire and Dario Izzo for helpful discussions.

References

[1] Murali Subbarao and Gopal Surya. Depth from defocus: a spatial domain approach. *International Journal of Computer Vision*, 13(3):271–294, 1994.

[2] Murali Subbarao. Parallel depth recovery by changing camera parameters. In *ICCV*, pages 149–155, 1988.

[3] Subhasis Chaudhuri, AN Rajagopalan, and S Chaudhuri. *Depth from defocus: a real aperture imaging approach*, volume 3. Springer New York, 1999.

[4] Takashi Nagata, Mitsumasa Koyanagi, Hisao Tsukamoto, Shinjiro Saeki, Kunio Isono, Yoshinori Shichida, Fumio Tokunaga, Michiyo Kinoshita, Kentaro Arikawa, and Akihisa Terakita. Depth perception from image defocus in a jumping spider. *Science*, 335(6067):469–471, 2012.

[5] MF Land. Structure of the retinae of the principal eyes of jumping spiders (salticidae: Dendryphantinae) in relation to visual optics. *Journal of experimental biology*, 51(2):443–470, 1969.

[6] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam, 2014.

[7] AD Blest, RC Hardie, P McIntyre, and DS Williams. The spectral sensitivities of identified receptors and the function of retinal tiering in the principal eyes of a jumping spider. *Journal of Comparative Physiology*, 145(2):227–239, 1981.