Investigation into the Influence of Biological Depth Cues on Monocular Depth Estimation for the Improvement of an Automated Privacy-Preserving Video Processing System

Research Topics

Jochem Groot Roessink
s2156059

Abstract Monocular depth estimation (MDE) in computer vision is the process of generating a depth map from a single 2-D image. This task is far from trivial since a 2-D image can be created from an infinite number of 3-D scenes. Fortunately, for many images captured in the real-world it is quite clear what the approximate depth is, and many MDE methods exist that produce results close to the ground truth. A depth map for an image has many use-cases, including the anonymisation of video material. Humans and some other animals can also get an idea about the depth of what they are observing when using only a single eye, for this they use so-called biological cues. An examples of these cues are the size of the observed objects and linear perspective. This work focuses on using prior knowledge about these biological cues to extract related information, which is used as extra input for an MDE. The goals are to determine the effect of these explicitly extracted cues on an MDE and to find out whether these explicit cues can be learned by an MDE instead, so that they are used implicitly. As a secondary contribution, a new data set containing RGB-depth image pairs is to be created.

Contents

Chapter 1

Introduction

This chapter describes the motivation for this study, as well as the research questions and requirements based on the motivation, and finally, a detailed description of biological cues, i.e. cues that are used by humans and/or other animals for depth estimation or monocular depth estimation (MDE) specifically. Chapter 2 provides a review of literature that is relevant to this study. For Chapter 3, initial experimentation has been done and the results are discussed for the extraction of biological cues. Chapter 4 describes the proposed method of the study, as well as the data sets and experiments that are needed to help answer the research questions using the method. Finally, Chapter 5 describes the planning.

1.1 Motivation

Video anonymisation is a useful tool for removing any private information from videos. An anonymised video can be stored, analysed and processed without being in conflict with any privacy regulation. Most anonymisation methods involve manipulating the original video in such a way so that no private information remains, an example is the blurring of faces. The problem with these kinds of methods is that when there is only a single object in a single frame that is not correctly recognised, and therefore not manipulated, the complete video is not anonymised anymore and it cannot be used. The longer a video, the greater the possibility that such a false negative occurs.

An alternative would be to reconstruct videos using default 3D objects. The advantage of this over the former would be that when a misrecognition occurs, the reconstruction might lose some accuracy but no private information is compromised in the result. Conde Moreno provided such a method [9]. This method can be summarised into the following steps:

1. Object detection per frame
2. Object tracking across frames
3. Depth estimation
4. Camera self-calibration
5. Object 3D location estimation (using steps 3 and 4)
6. Object trajectory smoothing (using steps 2 and 5)
7. 3D reconstruction using default 3D models

While most of these steps perform well on the used data set, the depth estimation performs quite poorly and can be seen as the bottleneck of the method. Al-jibouri provided an improvement for this step using a conditional Generative Adversarial Network (cGAN) [1]. This method shows promising results, but these results are still worse than the results of other methods (that used more data).

Because of these issues, it would be beneficial to improve upon this method. While simply adding more training data could improve the results, it might also be possible to improve the depth estimation on the same amount of training data by utilising prior knowledge. Humans and some animals utilise biological cues when estimating depth with a single eye, such as the size of objects and linear perspective (these cues are described in Section 1.3).

A neural network with enough capacity might be able to learn similar steps for producing a depth map. However, this could become quite complex: for example, learning about the size of many different objects using only unannotated images is far from trivial. While not impossible, this would likely require a lot more data than might be necessary. Therefore, alternatively, the known biological cues can be extracted from each image so that they can be used explicitly by the MDE system. In fact, Auty and Mikolajczyk have shown that explicitly providing the size of objects as extra input data using language models increases the performance of an MDE system [2].

A thorough search of the relevant literature has not yielded any work where any of the other biological cues have been used in a similar way. Therefore, there is novelty in studying the effect of explicitly using other biological cues, as well as using combinations of these cues. Besides that, Auty and Mikolajczyk did not compare a model without cues against one with cues for varying amounts of training data, so there is novelty in doing so as well. Although this last part might be challenging, since a lot of data with ground truth depth maps is required, it can help in determining two things. First, how much more efficient a model with cues is, by comparing the amount of training data at which performance between both systems is similar. Second, whether the model without cues is eventually able to match or even outperform its counterpart, which would indicate that similar steps to these biological cues are used by that model. Such an experiment could provide an interesting angle to the research.

1.2 Research Questions and Requirements

Research Questions The motivation describes the research that is to be done in this work, which leads to the following research questions:

- **RQ1**: How does the inclusion of biological cues affect the performance of an MDE system?
- **RQ2**: How does an MDE that utilises biological cues compare against the state-of-the-art?
- **RQ3**: How does the inclusion of biological cues affect the efficiency of an MDE in terms of training data?
- **RQ4**: How does the inclusion of biological cues affect the performance of an MDE for increasing amounts of training data?

Requirements As described in the motivation a lot of labelled data is required. To ensure enough available data can be used the following functional requirements are in place for the system:

- Should work for various image sizes
- Should work for a static single-camera set-up
- Should work for existing footage, so there is no control over camera movement or focal length

1.3 Biological Cues

This Section describes the biological cues that are used by humans and/or some animals to estimate depth. Besides that, it is described if and how these cues can be extracted from an image (or from a range of video frames) to be used as explicit input data for a depth estimation system.

(a) Linear Perspective (b) Foreshortening (c) Texture Gradient (d) Blur

(e) Motion Parallax (f) Overlapping (g) Light Scattering (h) Shading

Figure 1.1: Examples of different biological cues.

Binocular Vision The main depth cue that is used by humans is binocular vision. Since both eyes observe an object from a different position, a disparity exists between both observations of this object. A human can (intuitively) determine this approximate disparity which enables the sensation of depth [26].

A computer can approximate the same for an image pair captured- by a dual camera set-up. A pixel in one image can be matched to its corresponding pixel in the other image using a number of different techniques, an example is the method of Žbontar and LeCun [34]. Such a matching can be used to generate the disparity for every pixel. This, in combination with the relative orientation and distance of both cameras can then be used to estimate the depth in an image [27]. However, the required dual camera set-up does not align with the requirements.

Alternatively, when a single camera is moving and this movement can be tracked, then two frames could simulate a binocular vision system. Unfortunately, this would not work well for moving objects (people) and it would not work at all for a static camera set-up. Both of these are desired for this depth estimation system.

Another possibility might be to synthesise binocular image pairs from a single monocular image. Boer synthesised such binocular image pairs, but this method is not desirable, the reasons for which are stated in the literature review (Chapter 2) on Boer's thesis [4].

Object Size Besides binocular vision, there are other ways a person can get an idea about the depth of imagery they are observing. A good example of this is the size of observed objects. If multiple objects of the same type (for which it is known they are about the same size, for example, a collection of cars or a collection of people) are visible, and there is a clear difference in observed size, one can determine that the objects that look smaller are further away than the corresponding objects that look bigger, this will be called *relative size*. Additionally, an object of which the approximate real-world size is known can also guide the observer in estimating its depth, this will be called *known size*.

For a computer to extract this information an object recognition system can be used. Then, for each recognised object the observed position and size, as well as its class are determined. Having multiple objects in the same class already seems like sufficient explicit information for the relative size cue. Additionally, the mapping of a class to a known default real-world size provides sufficient explicit information for the known size cue as well.

Linear Perspective Another monocular depth cue is that of linear perspective. Lines that are parallel in the real world converge to a single vanishing point at the horizon when seen from a camera or eye. For an object consisting of parallel lines, the linear perspective cue tells what part of the object is closer to the camera and which part is further away. An example of an object that often consists of parallel lines is a building [25]. A visual representation of linear perspective can be found in Figure 1.1a. See Section 3.1 for a description of how a computer can extract this information.

Foreshortening This converging also applies to objects that do not contain parallel lines, for example when a person reaches their hand out to a camera, it is observed to be larger than the rest of their body, this effect is called foreshortening. Ivanov et al. showed that foreshortening can be used to determine the degree to which an object converges [19]. In Figure 1.1b the snout of the (right) cow is pointed at the camera, which makes it appear larger relative to the rest of its body, especially relative to its hind legs. Furthermore, there is a difference in size between its front and hind legs. Such cues can guide a human in determining the relative distance of each body part of the cow.

Observing this requires a lot of knowledge about any observed object and it might prove to be difficult to extract useful explicit information related to this effect. One possibility might be to recognise parts of an object (for example: hands of a human) as separate objects which can then be used by the same techniques that are used for extracting the object size.

Blur An eye or camera has a focal point. Objects closer to this point are perceived to be sharper than objects farther away from it, and the farther away an object is from this point, the more blurry it seems. In Figure 1.1d the focal point of the camera is near the woman and the tree she is leaning against, while the rest of the image is much farther away from it. The result is that the woman and the tree are much sharper than the rest of the image. A blur (or sharpness) map generated from the image could therefore provide useful information about the approximate relative distance of each part of the image [22].

The algorithm from Golestaneh and Karam produces exactly that: for each pixel, it takes a surrounding patch from the image and uses the magnitudes of high frequency waves extracted from the patch to determine as sharpness score for the pixel [16]. This algorithm is further discussed in Section 3.2.

Texture Gradient When an image includes a uniform texture gradient, a brick road for example, the coarseness of the texture can be used as a depth cue. As can be seen in Figure 1.1c, the grass is much more coarse and defined the closer it is to the observer, while the individual blades of grass unify into a single solid color the farther away they are from the observer [25].

To extract the coarseness from an image, the same algorithm used for the blur map can be used [16]. If there is a clear texture gradient present in an image, then the parts of it that are closer to the camera should receive a greater sharpness value than the parts that are farther away.

Accommodation In accommodation, the shape of the lens in an eye is altered in order to change the eye's focal length, during the observation of an object. Observing which (parts of) objects are in focus for each focal length can be used to estimate the relative distance. However, accommodation only works well for distances less than 2 meters [25]. Owls have been observed to use accommodation as a depth cue when viewing with a single eye [32].

This accommodation process of the eye could be replicated by a camera system. However, this would not align with the requirements. Alternatively, a collection of images with different focal lengths could be synthesised from a single image, but a system that performs this would require implicit depth knowledge, as well as the sharpening of blurred objects. The former would not make sense since it would require depth knowledge to later estimate the depth, and the latter is quite impractical as well.

Motion Parallax Humans and animals do not observe a static world. Objects in this world are moving. When these are moving in the same direction and at the same speed, objects closer to the observer seem to move faster. This works best for a moving observer that observes static objects, like in Figure 1.1e, where the bushes closer to the camera appear to move much faster than the trees in the background, which is visible in the image in the form of motion blur.

To incorporate motion parallax into a depth estimation system, one could simply add multiple frames of a video as input. Alternatively, an object recognition algorithm could track certain objects over several frames and this recorded movement could be used a explicit input data. However, since the system is required to work for a static camera set-up, it is not given that certain objects are supposed to move at the same speed and in the same direction, which would be required for the extracted data to be useful.

Overlapping If one object partly occludes another object, a human knows that the former is closer. In Figure 1.1f each cylinder except one is occluded by at least one other cylinder. Because of this, an observer can know that each cylinder has a smaller distance to the camera than the cylinders that it obscures. An object recognition system could extract this information, which can be used to determine which group of pixels are sure to be closer to the camera than another group of pixels.

Light Scattering When light from the sun hits the earth, some of it is scattered by molecules in the air. Some of this light gets directed towards a viewer observing an outdoor scene. When an object is very close to the viewer, by far most of the light observed is reflected from the object. The farther away an object is, the higher the probability that this scattered light is observed as well. This has the effect that the object's colour appears to be more similar to the colour of the sky. One could view it as the object being placed before the sky, and its opacity decreasing the further away it is from the viewer. An example of this effect would be mountain that is placed further away than other mountains which seems much bluer (Figure 1.1g). This information can be used a depth cue [10].

A computer could extract these cues from an image, by first recognizing the sky and its (average) colour. Afterwards, the image can be grouped into parts based on colour. Each group and its colour, as well as the colour of the sky, can then be used as explicit information for a depth estimation system. However, this method would also be prone to false positives, for example, any blue object would likely be recognised to be further away then it is. Combining this cue with other cues, such as overlapping, might improve results. Another issue, is that Al-jibouri's method uses a depth mask, so that the sky but also objects that are far away, like mountains, are discarded. Using this mask significantly improved the depth estimation [1]. Therefore, it would be desirable to incorporate a similar step for this project. Since light scattering has the greatest effect on far away objects, a trade-off between incorporating a depth mask and light scattering might have to be made.

Shading The shading of an object in an image provides useful information about its three-dimensional shape. In Figure 1.1f and Figure 1.1h the objects are in the shape of a cylinder. Due to the lighting, there is a visible gradient that gets darker the closer to the edge of the objects. This can guide an observer in understanding the 3D shape of an object, which can be used to determine the relative distance of each part of the object. Granrud et al. have shown that while 5-month-old infants do not use this cue, 7-month-old infants do [17].

Unfortunately, specific light conditions are required and the effect is only useful within an object, so extracting and using explicit shading information might be prone to errors and otherwise have no significant effect on the depth estimation.

Conclusion For many of the previously described biological cues, it is the case that they do not align with the requirements for the input data, or that their extraction is deemed impractical. The cues that will be further explored and for which an extraction system is/will be implemented are **object size**, **linear perspective**, **blur** and **overlapping**. As mentioned before, the extracted information related to object size and blur can also partly cover **foreshortening** and **texture gradient** respectively.

Chapter 2

Literature Review

This chapter is a review of the relevant literature, structured in the following segments: data sets, disparity, end-to-end MDE, cGAN methods and biologically inspired work. Additionally, a short conclusion is given.

Data sets To allow for the training and the evaluation of (M)DE systems, data sets containing RGB-depth pairs are needed. Saxena et al. provided such a data set, in which multiple scenes are recorded using both a standard camera and a laser scanner for the ground truth depth map [28]. Silberman et al. provided another data set, which is known as NYU Depth [29]. This data set contains 1149 of these RGB-depth pairs captured from 464 diverse indoor scenes, using a Kinect camera. Geiger et al. recorded traffic scenarios using multiple sensors, including stereo cameras and 3D laser scanners. Therefore, the resulting KITTI data set contains binocular RGB image pairs, paired with data, and consequently also monocular RGB-depth pairs. Because the recordings are from traffic scenarios, the data set consists of outdoor scenes. Eigen et al. proposed a specific train/test split for these last two data sets [11]. Mayer et al. provided another data set, which contains over 35000 synthesised stereo image pairs with a ground truth disparity [23]. As mentioned in Section 1.3, from the disparity the depth can be determined, which combined with either image from an image pair can form the desired RGB-depth pair. The data set from [8] includes over 20000 RGBD images of human poses recorded using a Kinect sensor [8]. However, due to the limited amount of scenes and actors, four and five respectively, only a limited amount of this data set can be used for training, otherwise, an MDE would likely be overfitted on those scenes.

Disparity When binocular image pairs are present, depth estimation essentially becomes a matching task: every pixel from one image should be matched. When each pixel from one is matched to a pixel in the other image, the pixel disparity can be determined, after which, estimating the depth map is a trivial process, as mentioned before (Section 1.3). Žbontar and LeCun provides such a method, it uses a CNN to determine the matching distance between two patches (one from each input image) to match pixels and eventually determine a disparity map [34].

Instead of explicitly matching pixels, end-to-end solutions exist that take a binocular image pair as input and output a disparity map directly. Godard et al. devised such a method. It uses a CNN that takes one image as input and produces both the left-right and right-left disparity maps. Using both of these disparity maps the left image can be reconstructed from the right image and vice versa. The differences between the input images and their reconstructed counterparts make up the cost function of the network, which leads to improvement of the disparity estimation during training. A depth map can be determined using either disparity map. Because this method only takes one image as input, it can be used to estimate depth for monocular images as well, although image pairs are still needed for training. The method does show plausible results for unseen data [15].

DispNet is another example of an end-to-end disparity estimation system. It consists of a CNN, modified from FlowNet, which is a network that uses a correlation layer to estimate optical flow between two images [12]. The modifications made for DispNet allow for the estimation of disparity instead [23].

The main disadvantage of disparity methods is that binocular image pairs are required, which is not in line with the requirements (Section 1.1) for this work. While the method of Godard et al. is able to produce

results for unseen monocular image data, it did not perform well in the work of Conde Moreno [9], and a method that can be trained on monocular images is likely to perform better.

End-to-end MDE Instead of estimating the disparity from an image pair (or a single image), there are methods for generating depth from an RGB image directly. One way is to group neighbouring pixels into so-called 'superpixels' of which the 3D orientation can be estimated so that the combined orientation of all superpixels can be used to create a depth map for the input image. This is what the method from Saxena et al. does, however it does not perform well compared to later work [28].

In this later work, the focus lies on training a neural network on RGB-depth pairs directly. Such a network takes an RGB image as input and produces a depth map of the same resolution, the differences between this produced depth map and the ground truth depth map make up the cost function that ensures improvement of the depth map estimation.

Eigen et al. provided such a method, in particular one that uses two CNNs. The first CNN generates a complete depth map for a low-resolution version of the input image, while the second CNN takes both this depth map and a smaller patch of the input image as input. The details in this patch, like wall edges, should improve upon the original fine depth estimation. While the method outperformed older methods that only acted as a baseline, the method is not compared to other methods. Luckily, this issue is overcome by the fact that the train/test split from this work is used by later work as well, which allows for comparison [11].

The method from Laina et al. uses a fully convolutional architecture. This method has better performance compared to the method from Eigen et al. for every metric except for one, for which the performance is equal [21].

Bhat et al. provided a method where an image is encoded using the EfficientNet CNN [31]. The encoding is decoded into a feature map for every pixel position, which is fed into another CNN which produces the final depth map. This method outperforms both the method from Laina et al. and the one from Eigen et al. for every metric used [3].

cGAN Methods A cGAN consists of two networks: a generator and a discriminator. The generator works similarly to the previous methods, the discriminator receives both a true depth map and one generated from the corresponding image by the generator and its task is to determine which is which. The cost function has to goal to ensure that while the discriminator gets better at distinguishing true depth maps and generated ones, the generator gets better at creating depth maps.

Zhang et al. and Chen et al. both provided such a method, and both outperform the method from Laina et al., and consequently, the method from Eigen et al. in every metric used [33, 7]. Similarly, for their thesis, Al-jibouri used a cGAN to estimate depth, however in this case the used method was not able to get better results than Eigen et al., but it is noteworthy that the latter is trained on more data. The use of more data might further improve the performance of this method [1]. The thesis of Al-jibouri is also the work from which this study directly follows.

Alternatively, a cGAN could also be used to generate binocular image pairs from a single image. These pairs can then be used in a disparity method, to finally estimate the depth. However, as described before, estimating this disparity is not a trivial process and can be prone to errors. Therefore, compared to estimating disparity or depth directly, first converting single images to a stereo image pair and then estimating depth only seems like a way to increase noise. Boer provided such a method. While this work showed that such a method can be used to estimate depth, it was indeed not able to outperform the state-of-the-art [4].

Biologically Inspired Work Most previous biologically inspired computer vision work is focused on replicating the behaviour of certain cells, with the goal of achieving similar functionality to the observed functionality of those cells. This idea is similar to what led to the invention of the perceptron [24], which was inspired by the behaviour of a single neuron and formed the basis for neural networks.

Similarly, it has been suggested that simple cells in the visual cortex can be modelled by 2-D Gabor filters [13]. Furthermore, convolutional kernels learned by the first convolutional layer of a CNN trained on pictures seem to resemble such Gabor filters [20]. This would explain the use of CNNs in computer vision tasks, the popularity of which is emphasised by the fact that much of the previously described work makes use of CNNs.

However, regular CNNs still have limitations, an example of which is their robustness against unseen noise. Strisciuglio et al. created a new layer for CNNs, which is inspired by the push-pull inhibition phenomenon, a phenomenon that is observed in certain neurons in the visual cortex. Replacing the first layer of existing CNNs with this push-pull layer, significantly enhanced the robustness against noise that was unseen during training [30].

Not all biologically inspired computer vision work is about replicating phenomena observed in neurons in the visual cortex, Chen et al. discussed the use of an event-based neuromorphic vision sensor for autonomous driving. In contrast to regular cameras, this sensor does not record and process the brightness for every pixel for every frame, but observes brightness changes and triggers asynchronous events based on that, which is more similar to how human eyesight works. The advantages of this over regular cameras include low energy consumption, high dynamic range and no motion blur problems [6].

In regards to MDE, Auty and Mikolajczyk provided an adaptation to a scaled-down version of the method from Bhat et al. [3]. Instead of using just a three-channel (red, green and blue) image, their work explicitly extracts information about both the known size and relative size of objects and encodes this into extra channels for the input image. This method has not been able to achieve better performance than the original method from Bhat et al., but the latter has a significantly greater capacity. In fact, this method does achieve significantly better performance than the scaled-down version it is directly based on, for every metric. Therefore, this work has shown that the addition of explicit biological cues to the input space can improve the performance of a monocular depth estimation system.

Conclusion This literature review describes and compares different methods for MDE, each having different advantages and disadvantages. Therefore, the described studies provide useful insights for this work. Additionally, the biologically inspired work shows how taking inspiration from biological vision systems can have a positive impact on the development of computer vision methods. Most relevant to this work is the work of Auty and Mikolajczyk, which shows that the explicit addition of a biological cue, namely information about the size of objects, can improve the performance of an MDE system [2]. What has not been addressed in the literature above, nor has it been found in any of the other relevant literature, is a study on whether the addition of other known biological cues can also improve the performance of an MDE system. Therefore, what will be addressed in this work (as mentioned in the motivation, Section 1.1) will bring novelty.

Chapter 3

Preliminary Experiments

This chapter describes the initial experimentation for the extraction of two biological cues, namely linear perspective and blur.

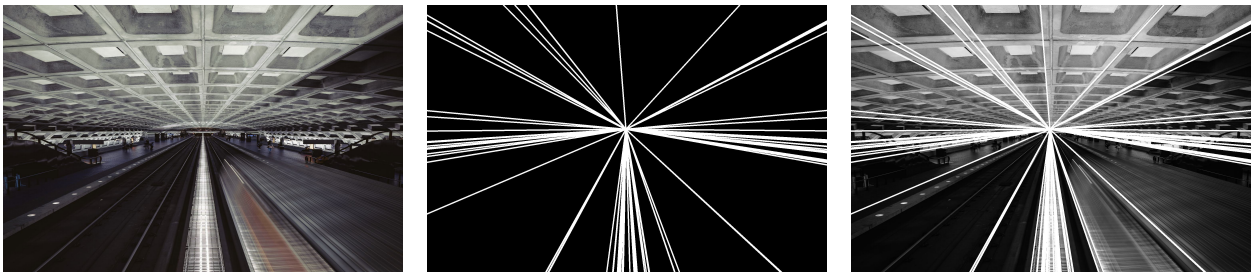
3.1 Linear Perspective

Method To extract cues from an input image and encode them as another image of the same resolution the following methods are used. The Canny edge detection algorithm [5] is used to find significant brightness steps in the blurred, grey-scale input image in any direction, which are the desired edges. Afterwards, a Hough transform [18] is used on this edge map to find line pieces that are of a minimum desired length and have a maximum gap size. For every pair of line pieces, the intersection point of their corresponding mathematical lines is determined. To estimate the vanishing point, the average of most of these points is calculated: for both x and y, the values are ranked, and the weighted average is calculated from the centre 70% (since two lines that are nearly parallel and are spaced far apart can have an intersection point that has such high values that it has a significant influence on the result), with the weight being the product of the lengths of the two line piece each point is the intersection of. The line pieces that do not intersect this vanishing point within a certain margin are filtered out. Finally, new line pieces are created from the remaining ones, that start from the vanishing point and intersect with the furthest point end of each line piece, until the border of the image. This generates the result that can be found in Figure 3.1

Discussion The resulting image in Figure 3.1 looks like a useful cue for estimating depth. A human can use it to get an idea of where the vanishing point is, as well as significant edges, and have an idea about the depth of different parts of the image. However, this method depends on a lot of input constants, like minimum line length, amount of blur and the intersection margin for filtering out line pieces. If the incorrect values for an image are used, no lines or too many that are not related to the linear perspective might be detected. Solely basing these values on the size of an input image has not been successful yet, and to generate desirable results, setting them manually is still required for most images.

An optimiser might have to be used that checks for several input values and finds the best ones for each image, however, this can have quite a negative effect on the time performance. Additionally, the determination of all intersection points has a time complexity of $\mathcal{O}(n^2)$ for n line pieces, which does not scale well at all. Luckily, when an optimiser is used, the number of line pieces should remain relatively low.

Furthermore, while setting the values manually for images that show a lot of linear perspective creates desirable results, for other images it is more difficult since there can be either too many lines that are detected that are not related to the linear perspective and/or too many of the lines that are related that are filtered out. If the method can be improved so that it works for most images that show some linear perspective and does not generate any lines for images that do not, the extracted cue can be useful. This is because there are existing depth data sets containing road data [14], where the lines of the road as well as its edges can show linear perspective. Experiment 0 (Section 4.3) can be used to find out whether the extracted cues

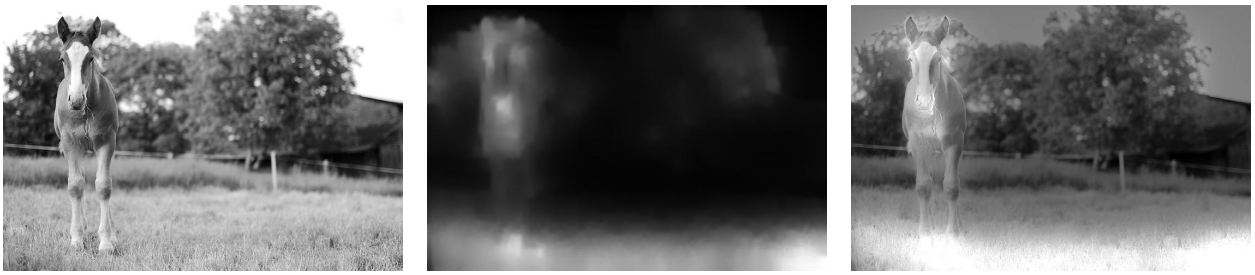


(a) Input image that contains a great amount of linear perspective.

(b) The lines extracted using the described linear perspective method.

(c) The same lines added on top of the input image

Figure 3.1: Qualitative result of the linear perspective experiment.



(a) Input image of a horse with a blurry background.

(b) The extracted sharpness map, the intensity corresponds to sharpness.

(c) The previous two images blended together.

Figure 3.2: Qualitative result of the blur experiment.

are actually useful for MDE and/or whether using a circular gradient with the vanishing point in the center might be better.

3.2 Blur

Method As described in Section 1.3 it would be desirable to obtain a blur map for every input image, which can be used as extra input for an MDE system. The method from Golestaneh and Karam achieves exactly that. It is based on the fact that a grey-scale image can be transformed into a sum of waves for both the vertical and horizontal direction (in this case, using a discrete cosine transform). For a blurry image, the amplitudes of the resulting waves with high frequencies are on average lower than for a sharper image. For every pixel, a patch of a certain size in which the pixel is in the centre is taken from the original image, and the amount of blur is determined using the coefficients of the higher frequency waves obtained using the discrete cosine transform. This is done for the original image, but also for the image blurred at different scales, which together can be used to determine the final amount of blur for each pixel. Additionally, the results are normalised so that the pixel with the highest sharpness value has value of 1 and the pixel with the lowest 0. A complete description of the method can be found in the original paper [16].

Discussion Qualitative evaluation of the method resulted in satisfactory results. As can be seen in Figure 3.2, the horse and the grass close to the camera generally get high sharpness values, while the blurry background gets much lower sharpness values. Furthermore, the grass can be seen as a texture gradient, and the algorithm correctly detects that it is more coarse closer to the camera. It is true that some parts of the resulting map do not seem realistic, like how the white part of the horse's head gets a lower sharpness estimation than the rest of the head. However, using just the blur map can already be used by a human to have an idea about the depth in the image. Experiment 0 (Section 4.3) can confirm whether this cue is also useful for machine MDE. Additionally, for texture gradient the higher coarseness is always closer to the camera, but for blur it is possible that an object is closer to the camera than the focal point, so it can be

more blurry than an object that is further away. However, this is mostly present in professional photographs and manual inspection of public depth data sets found no occurrence of such a phenomenon, so it can be assumed that for most pictures, blur can be positively related to depth. The main downside of this method is the time performance, it scales up quadratic on the image size. For relatively small images the time performance is acceptable, but if a data set contained many larger images, it can be problematic. If this is the case, there are two possibilities: scaling down an image that is too large and scaling back up the result, or using a modified variant that has better time performance. The former will surely decrease the accuracy and the latter likely results in decreased accuracy as well.

Chapter 4

Proposed Method

This chapter describes the proposed method for this study, as well as the experiments that should help in answering the research questions.

4.1 Model

The proposed schematic can be found in Figure 4.1. This work directly follows from the thesis of Al-jibouri, whose end-to-end MDE method used a cGAN [1]. The generator receives an RGB image as input and learns to produce a depth map of the same dimensions. The input for the discriminator is an image paired with either the generated depth map or the ground truth depth map and learns to decide which is which. The generator and discriminator from this method can be found in Figure 4.2. The proposed model is based on this, but with a modified generation part, so that it allows for the inclusion of explicit biological cues. The main modifications to be applied to the generator and discriminator are the following:

- Larger image width and height (for example 480×960) so that more image sizes are supported.
- Greater amount of image channels (from 3 to at least 8) to allow for the addition of (different combinations of) cues to the input space.
- Further modifications that ensure that the generator actually uses the additional channels and not just the color channels.
- Further modifications that ensure that the generator and the discriminator find an equilibrium.

The reasons for choosing a cGAN over a traditional CNN architecture are the following:

- Zhang et al. and Chen et al. have shown that a cGAN is able to outperform the depth estimation of other methods (including ones that use a more conventional CNN) in every metric used.
- While Auty and Mikolajczyk have shown that the addition of explicit biological cues can improve the performance of a more conventional CNN [2], a thorough search of the relevant literature has not yielded any similar work for a cGAN. Therefore, there is novelty in using a cGAN.

The reasons for modifying the input space over another possibility like encoding cues in a hidden layer are the following:

- While the chosen biological cues are known to be used by humans, most of them cannot be linked to a specific phenomenon observed in a specific group of cells in the visual cortex, as far as is known. This means that there is nothing a hidden layer can be modelled after.
- Existing methods that can be used to extract cues, like the method for extracting blur (Section 3.2), already encode the extracted information as a map of the same resolution as the input image [16]. Additionally, Auty and Mikolajczyk have shown that object size cues can also be encoded into multiple of these channels [2].

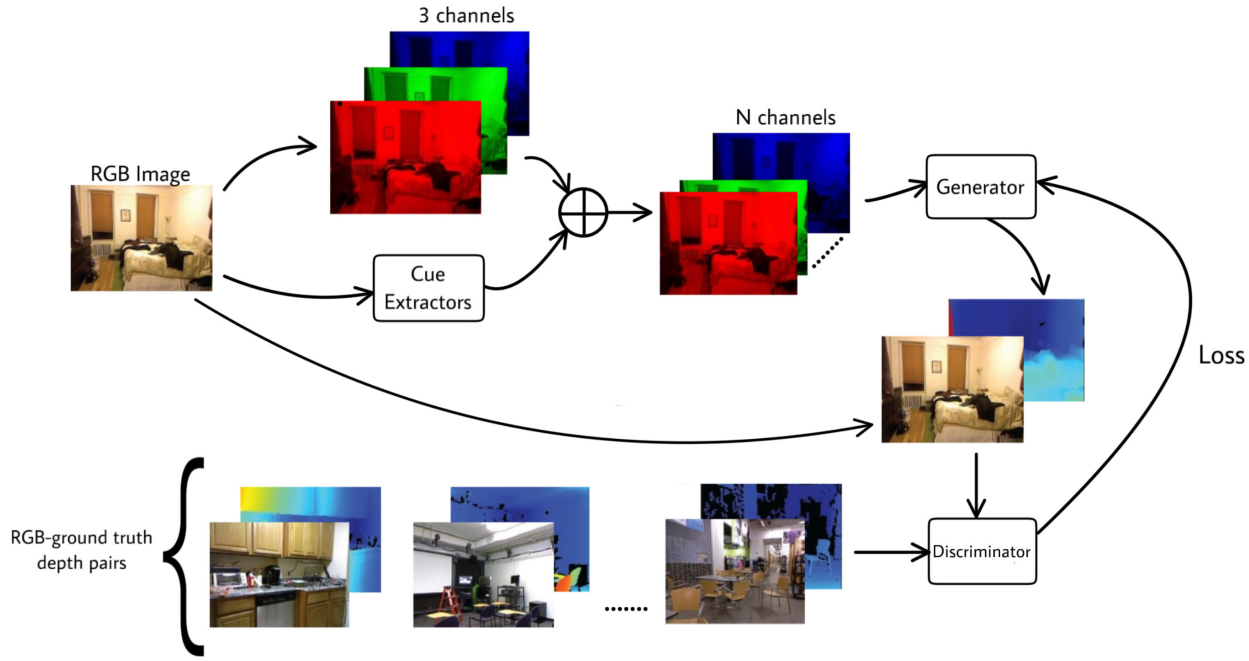


Figure 4.1: Schematic of the proposed model, based on the schematic of Chen et al. [7], with extensions that enable the use of biological cues. The RGB and depth images are from NYUDepth [29].

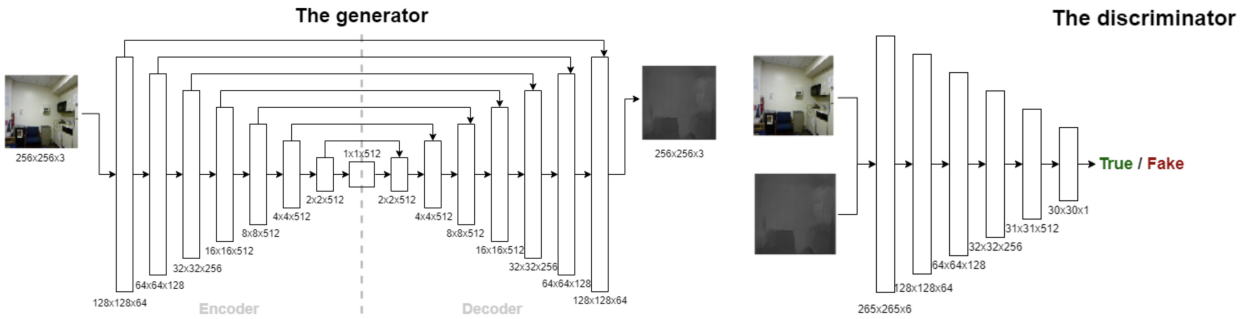


Figure 4.2: The generator and discriminator from Al-jibouri [1].

- There is a better comparison between the model that is trained on only input images (control model) and when it is trained with additional depth information, since the architecture is the same.
- Combining multiple cues is trivial since all generated channels can be stacked upon the original image channels.

4.2 Data sets

As described above the proposed model requires RGB-depth pairs for both training and evaluation. There are several existing data sets that already contain this desired data format, or that can be used to generate the desired data format, these are: Make3D [28], KITTI [14], NYUDepth [29], DispNet [23] and FlyingChairs [8]. These are described in more detail in the literature review. Additionally, other data sets that contain data that already are or can be transformed into RGB-depth pairs, could also be used.

Contribution To ensure the presence of enough training data, it might be necessary to create a new data set, work on this will be done before the actual work on the thesis and the implementation of the model. One possibility is to generate RGB-depth pairs using 3D data from hyper-realistic video games, like Grand Theft Auto 5 or Red Dead Redemption 2. Additionally, this data can be recorded using a camera setup that consists of both an RGB camera and a Lidar scanner, which are present in the recent, high-end iPhone and iPad models. If one or both of these methods is deemed successful in creating usable data, a new data set can be created, which can be considered an extra contribution of this work.

4.3 Experiments

To use the model previously described to answer all the research questions the following experiments are to be performed, each of which uses a certain subset of the collection of data sets described above. All of these should be performed in the given order, except for Experiment 3 and Experiment 4, which can be performed in parallel.

For every experiment a model is trained that produces a depth map, for which a ground truth depth map exists. Therefore, evaluating the performance of a model consists of measuring the difference between the estimated depth for a pixel and its ground truth value. For a pixel collection $\mathcal{P}$ consisting of ground truth depth values p paired with estimated values $\hat{p}$, the following metrics can be used (which also have been used in other work [11, 1]):
Thresholded difference: % of pixels for which $\delta = \max(\frac{p}{\hat{p}}, \frac{\hat{p}}{p}) <$ threshold
Absolute relative difference: $\frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \frac{|p - \hat{p}|}{p}$
Squared relative difference: $\frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \frac{(p - \hat{p})^2}{p}$
Linear RMS Error: $\sqrt{\frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} (p - \hat{p})^2}$
Log RMS Error: $\sqrt{\frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} (\log_{10} p - \log_{10} \hat{p})^2}$
Scale-invariant log RMS Error: $\sqrt{\frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} (\log_{10} \hat{p} - \log_{10} p + \alpha(\mathcal{P}))^2}$
where $\alpha(\mathcal{P}) = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} (\log_{10} p - \log_{10} \hat{p})$

Experiment 0: Usefulness of each Cue

For this preliminary experiment, an extracted cue is used as the only input for an MDE system. It can likely be performed using a relatively small subset of the available data. This would allow for the determination of which of the cue extraction methods can be used by the MDE system to significantly outperform random estimation. Besides that, this experiment can also be used to select the best out of a group of candidate methods for one of the depth cues.

Experiment 1: Performance of each Cue

For each chosen cue, the extraction methods are applied to the same input images, and a model that utilises the cue is trained on these images. Additionally, a model that does not utilise any cues (control) is trained on the same images. After training, all models are evaluated on the same group of test images. This experiment likely requires a larger subset of the available data to be performed. The results from this should help with determining whether depth cues can increase the performance of an MDE method (**RQ1**), and how the different depth cues compare against each other.

Experiment 2: Performance of Combined Cues

Given that they exist, the cues that significantly outperform the control model are all combined and used to train a model on the same training set from Experiment 1. It is also evaluated on the same test set so that it can be compared to the models from Experiment 1. This result should help in determining whether combining multiple depth cues can further improve results (given that more than one depth cue can individually improve depth estimation), which is again related to **RQ1**.

Experiment 3: Performance of Model with Cues compared to the State-of-the-art

The model from Experiment 1 and Experiment 2 with the lowest error, is compared trained and evaluated using a specific subset of the available data, namely, the Eigen split [11]. Since the training and test sets are used in other work this allows for a comparison of the best model from this work against the state-of-the-art, which should help in answering **RQ2**.

Experiment 4: Investigation into the Effect of Cues on the Efficiency and Performance of a Model in terms of Training Data

Two models are used from Experiment 1 and/or 2: the one with the best performance and the control model. Given that the former significantly outperforms the latter on relatively small amounts of data, both models are trained on an increasing amount of data. First, these results should allow for the comparison between a model with cues trained on a lower amount of data and a model without cues trained on a higher amount of data, which helps in answering **RQ3**. Furthermore, they should indicate whether the performance of the control model gets closer to that of the other model for an increasing amount of training data and if it eventually matches that model, or even outperforms it (**RQ4**). This could indicate that a model without explicit cues eventually learns to use similar steps.

Chapter 5

Planning

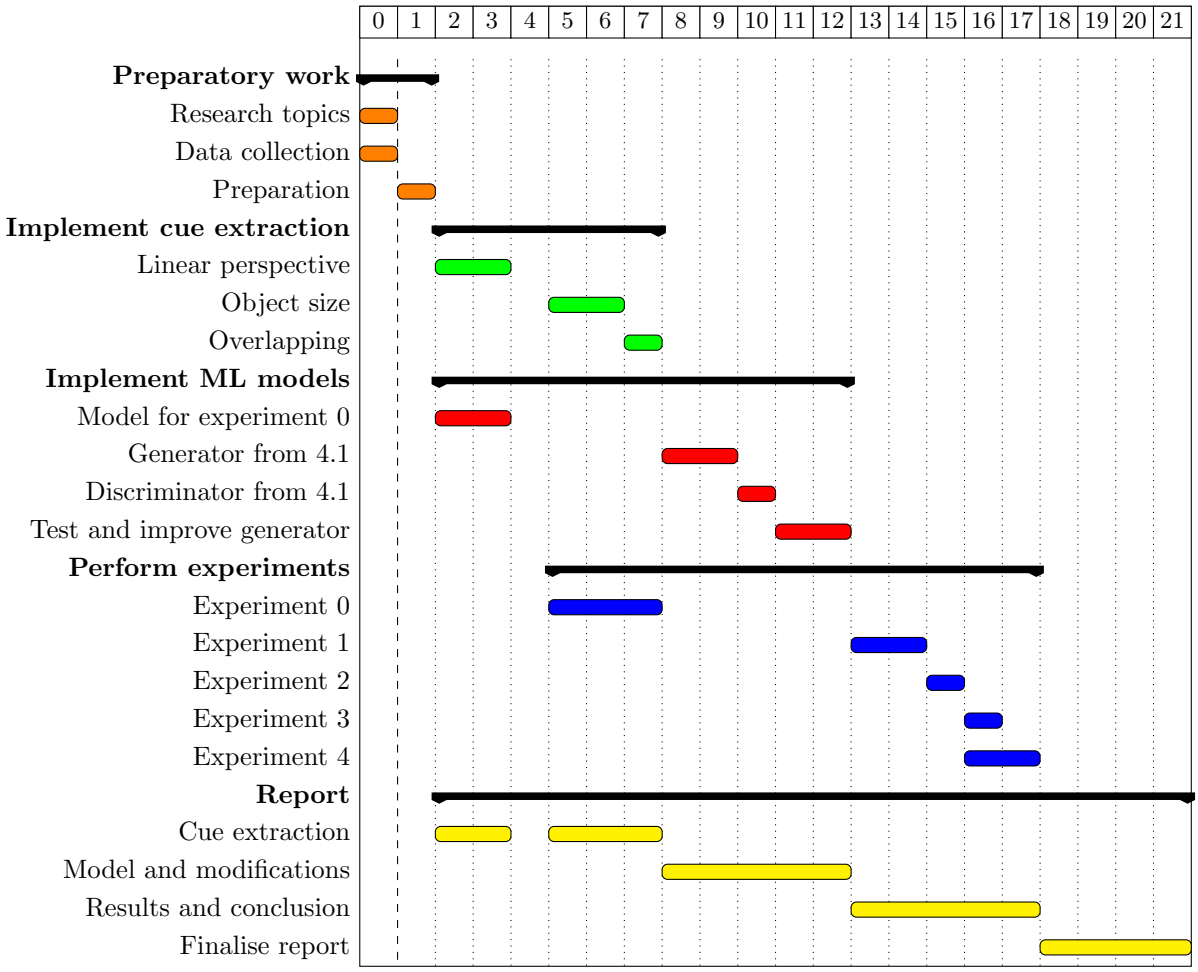


Figure 5.1: Weekly planning. Week 1 is the week of Monday 6 February 2023 and week 21 of Monday 26 June 2023. The index 0 refers to the period from September 2022 until week 1.

An overview of the planning can be found in Figure 5.1. This is further elaborated below.

Data Collection This task is performed during the period between the research topics and work on the actual thesis, the goal is to gather enough data for Experiment 4. This period lasts about three months, which might be considered to be a lot for just data collection, but since this is before the actual thesis, work this work will be part-time, so three months seems realistic.

Cue Extraction During the actual work on the thesis, the first step is to will extract each selected biological cue (Section 1.3). For the blur cue, there already is a method for extracting it in the desired form, some work might have to be done in creating a similar method with better time performance, but since the images from most public data sets have acceptable sizes, this is not expected. For the linear perspective cue, some work has already been done as well, however, improvement is still needed to make it work for most images that show linear perspective. Extracting the object size cue is expected to take the most time, since an object recognition algorithm has to be selected, which classes are used have to be determined, average object sizes have to be determined for each class and all of this information has to be encoded into the desired format. Luckily, Auty and Mikolajczyk already worked on this, which provides some insight. Extracting the last cue, overlapping, it is not expected to take much time, since the same object recognition algorithm can be used.

Model The following step is the implementation of the model from Section 4.1. After it is determined which cues can be used and how many channels each cue requires, the generator from Figure 4.2 can be modified to allow for the amount of image channels needed. Additionally, the other modifications mentioned in Section 4.1 are applied to the model. Implementing this model and testing that the model actually uses the additional image channels and the generator and discriminator train to find an equilibrium is expected to take around five weeks.

Experiments The first experiment that is performed is Experiment 0, to determine which of the extracted cues can be used, this experiment can be performed before the final model is implemented. After the model is implemented, the other experiments are used to evaluate it. Doing this and discussing the results is expected to take around five weeks. This is mainly because every experiment is dependent on the results of prior experiments, except for the last two experiments, which can be performed in parallel. Besides, an available environment has to be found, and training the model for each experiment is expected to take quite some time, especially for Experiment 4.

Report During the implementation, what will be worked on, will be described in the report. During the experimentation the results and the conclusion will be written. Finally, the last four weeks consist of cleaning up and finalising the report. Additionally, this last period also functions as a buffer for work that takes longer than originally planned.

Bibliography

[1] Zina Al-jibouri. Improving depth estimation in an automated privacy-preserving video processing system. Master's thesis, Radboud University, Nijmegen, NL, April 2020.

[2] Dylan Auty and Krystian Mikolajczyk. Monocular depth estimation using cues inspired by biological vision systems. arXiv preprint arXiv:2204.10384, 2022.

[3] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. Adabins: Depth estimation using adaptive bins. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 4009–4018, June 2021.

[4] Sverre Boer. Depth estimation on synthesized stereo image-pairs using a generative adversarial network. Master's thesis, University of Twente, Enschede, NL, July 2021.

[5] John Canny. A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-8(6):679–698, 1986. doi: 10.1109/TPAMI.1986.4767851.

[6] Guang Chen, Hu Cao, Jorg Conradt, Huajin Tang, Florian Rohrbein, and Alois Knoll. Event-based neuromorphic vision for autonomous driving: A paradigm shift for bio-inspired visual sensing and perception. IEEE Signal Processing Magazine, 37(4):34–49, 2020. doi: 10.1109/MSP.2020.2985815.

[7] Richard Chen, Faisal Mahmood, Alan Yuille, and Nicholas J. Durr. Rethinking monocular depth estimation with adversarial training, 2018. URL `https://arxiv.org/abs/1808.07528`.

[8] Christian Dornhege Wolfram Burgard Christian Zimmermann, Tim Welschehold and Thomas Brox. 3d human pose estimation in RGBD images for robotic task learning. In IEEE International Conference on Robotics and Automation, ICRA, 2018. URL `https://lmb.informatik.uni-freiburg.de/projects/rgbd-pose3d/`.

[9] Lucía Conde Moreno. Automated privacy-preserving video processing through anonymized 3d scene reconstruction. Master's thesis, Utrecht University, Utrecht, NL, September 2019.

[10] F. Cozman and E. Krotkov. Depth from scattering. In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 801–806, 1997. doi: 10.1109/CVPR.1997.609419.

[11] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network, 2014. URL `https://arxiv.org/abs/1406.2283`.

[12] Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks, 2015. URL `https://arxiv.org/abs/1504.06852`.

[13] Itzhak Fogel and Dov Sagi. Gabor filters as texture discriminator. Biological cybernetics, 61(2):103–113, 1989.

[14] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. The International Journal of Robotics Research, 32(11):1231–1237, 2013.

[15] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency, 2016. URL https://arxiv.org/abs/1609.03677.

[16] S. Alireza Golestaneh and Lina J. Karam. Spatially-varying blur detection based on multiscale fused and sorted transform coefficients of gradient magnitudes, 2017. URL https://arxiv.org/abs/1703.07478.

[17] Carl E Granrud, Albert Yonas, and Elizabeth A Opland. Infants' sensitivity to the depth cue of shading. Perception & Psychophysics, 37(5):415–419, 1985.

[18] J. Illingworth and J. Kittler. A survey of the hough transform. Computer Vision, Graphics, and Image Processing, 44(1):87–116, 1988. ISSN 0734-189X. doi: https://doi.org/10.1016/S0734-189X(88)80033-1. URL https://www.sciencedirect.com/science/article/pii/S0734189X88800331.

[19] Iliya V. Ivanov, Daniel J. Kramer, and Kathy T. Mullen. The role of the foreshortening cue in the perception of 3d object slant. Vision Research, 94:41–50, 2014. ISSN 0042-6989. doi: https://doi.org/10.1016/j.visres.2013.10.019. URL https://www.sciencedirect.com/science/article/pii/S0042698913002617.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, Advances in Neural Information Processing Systems, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[21] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks, 2016. URL https://arxiv.org/abs/1606.00373.

[22] George Mather and David R R Smith. Blur discrimination and its relation to blur-mediated depth perception. Perception, 31(10):1211–1219, 2002. doi: 10.1068/p3254. URL https://doi.org/10.1068/p3254. PMID: 12430948.

[23] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4040–4048, 2016.

[24] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biology, 52:99–115, 1990.

[25] Takanori Okoshi. Three-dimensional imaging techniques. Elsevier, 2012.

[26] Gian F Poggio and Tomaso Poggio. The analysis of stereopsis. Annual review of neuroscience, 7(1): 379–412, 1984.

[27] Pablo Revuelta Sanz, Belén Ruiz Mezcua, and José M Sánchez Pena. Depth estimation-an introduction. In Current Advancements in Stereo Vision. IntechOpen, 2012.

[28] Ashutosh Saxena, Min Sun, and Andrew Y Ng. Make3d: Learning 3d scene structure from a single still image. IEEE transactions on pattern analysis and machine intelligence, 31(5):824–840, 2008.

[29] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In European conference on computer vision, pages 746–760. Springer, 2012.

[30] Nicola Strisciuglio, Manuel Lopez-Antequera, and Nicolai Petkov. Enhanced robustness of convolutional networks with a push–pull inhibition layer. Neural Computing and Applications, 32(24):17957–17971, 2020.

[31] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020.

[32] Hermann Wagner and Frank Schaeffel. Barn owls (tyto alba) use accommodation as a distance cue. Journal of Comparative Physiology A, 169(5):515–521, 1991.

[33] Wei Zhang, Guoying Zhang, and Qiran Zou. Depth prediction from monocular images with cgan. In International Conference on Smart Computing and Communication, pages 427–436. Springer, 2018.

[34] Jure Žbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. 2015. doi: 10.48550/ARXIV.1510.05970. URL https://arxiv.org/abs/1510.05970.