The Plymouth OWL, a new open-source robot for stereo depth using vergence and disparity

Rogers J, Page-Bailey K and Culverhouse PF,
Centre for Robotics and Neural Systems,
Plymouth University, Plymouth, PL4 8AA, UK

Abstract

A new Raspberry Pi-based robot is presented that allows exploration of stereo vision algorithms. Depth information of a target is found via two methods, vergence and disparity. Vergence requires physical tracking of the target, moving each camera to align with a chosen object, and through successive camera movements (saccades) a sparse depth map of the scene can be built up. Disparity however requires the cameras to be fixed and parallel, using the position of the target within the field of view, of a stereo pair of cameras, to calculate distance. As disparity does not require the cameras to move, multiple targets can be chosen to build up a disparity map, providing depth information for the whole scene. In addition, a salience model is implemented imitating, at a simple level, how people explore a scene. This is achieved with feature maps, which apply filtering to the scene to highlight areas of interest, for example colour and edges. This is purely a bottom-up approach, and is based on Itti and Koch's saliency model (Itti and Koch 2000).

Introduction

Animals have an amazing ability to quickly and efficiently extract useful information from their environment. Humans for example, can react to visual stimuli with a mean reaction time of 180 to 200 milliseconds (Shelton and Kumar, 2010). To explore how this could be done, a robotic analogue for the human visual system has been developed at Plymouth to assess algorithms, and begin to mimic observed human eye behaviours.

The Owl robot is a stereo camera host designed for the exploration of verging camera stereopsis. It has five degrees of freedom offering Neck rotate with Stereo eyes with pan and tilt. Local processing is by the Raspberry Pi dual camera compute board located at the base of the robot. An additional PCB provides an interface to the Pi for Servo control and an audio codec. The cameras are Pi HD cameras set to deliver stereo pairs at VGA resolution and streamed using RTP protocol web streaming at 30fps. See Fig. 1 for a photo of the robots, camera separation is 65mm.

A pair of MKS DS65k high-speed digital servos moves the eyes with a 333Hz period. Normal drive Pulse Width Modulation (PWM) is 3ms period, with a pulse width between 850us- 2150µs (ie. 1300 PWM value range). The centre position is set at approximately 1,500µs and the servos have a dead band of one microsecond. The PWM range allows for 160° rotation, with one PWM step being 0.113°.

The Pi compute board was chosen as it offers dual CSI format (DMA) camera inputs, that facilitate the high-speed streaming of camera images to the internet. The processor is BCM2835 (the same installed in Raspberry Pi B+).

Figure 1: A parliament of Plymouth OWL robots

The eyes of the robot are OV5647 camera modules, each capable of 2592 x 1944 pixels in a 4:3 aspect ratio. The cameras have been set to display a lower 640 x 480 pixels, as higher resolutions are not required and this reduces image processing time. The two video streams are combined into a single 1280 x 480 stream, which is then communicated to a host computer via an RTP interface over USB in the MJPEG format (UV4L format – see linuxprojects.org). They are not synchronised at present. The cameras can pan and tilt independently via the four high-speed MKS DS65K servos, each capable of a no-load angular velocity of 0.203 sec/60° at 4.8V (source: www.mksservosusa.com). This equates to approximately 300°s^{-1}. Observational data has shown that human eye saccades average 160°s^{-1} (Abrams et al., 1989), but can reach a peak angular velocity of 900°/s (Wilson et al., 1993), thus the robotic analogue cannot capture the full speed of the human visual system. However, 300°/s is satisfactory for the experiments planned.

The neck of the robot can rotate about one axis with a Corona DS558HV servo, offering a range of 160 degrees and a peak no-load angular velocity of 300°s^{-1} (source: hobbyking.com). The velocity will be lower due to the mass of the owl head, but still satisfactory for target tracking experiments. Servos are controlled by the Pi compute module using PiGPIO module (see abyz.co.uk/rpi/pigpio), the PWM position values are generated by the host computer. The Pi compute module runs an IP server program (available as either python script or C-code program) which creates an IP socket over the host USB connection using the TCP protocol. The script waits for a 24-byte packet, which contains five 4-digit decimal integer numbers in an ASCII string separated by spaces, these are the new servo positions instructed by the host computer.

Software limits are applied to these new positions so that the servos do not over actuate to a state where the cameras or cables are damaged.

The host computer supports the main software, programmed in C++ with the OpenCV 3.2 library. This arrangement allows for faster computations, compared to just using the on-board Pi compute module, and offers the potential to use GPU arrays for vision and deep learning in the future. Cameras were calibrated using OpenCV functions to correct for intrinsic distortions.

Vision Depth approximation of a single target (vergence)

When focusing on a target, both eyes move to centre the target about each fovea (a 64x64 patch at the centre of the field of view of each camera). This behaviour is known as Vergence, and provides depth information via the trigonometric relationship between the angles of the eyes and their distance from each other.

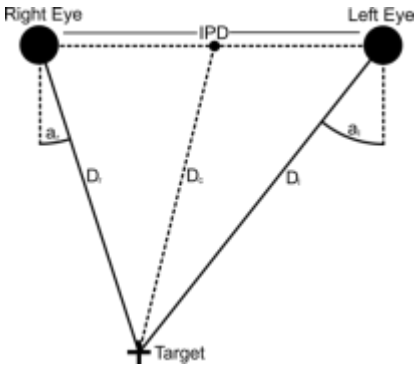

Figure 2: Parameters needed to find distance to a target, using vergence.

Figure 2 better describes the geometric problem. The known variables are a_r, a_l and Inter-Pupillary Distance (IPD). From this, D_c is to be calculated, which is the distance to the target from the centre position of both eyes. To find this distance, all internal angles of the triangle drawn from both eyes to the target have to be calculated. Assuming angles are measured clockwise, the internal angles of the eyes are expressed in equations 1 and 2:

$$Right\ Internal\ Angle = 90 + a_r \quad [Eqn.1]$$

$$Left\ Internal\ Angle = 90 - a_l \quad [Eqn.2]$$

As all angles of a triangle equate to 180 degrees, the angle at the target, created by the alignment of the eyes, is (Equation.3):

$$Target\ Angle = 180 - (90 - a_l) - (90 + a_r)$$

$$Target\ Angle = a_l - a_r \quad [Eqn.3]$$

Using the sine rule, the distance from the eyes to the target can be found, for example the left eye (Equation.4):

$$\frac{\sin(a_l - a_r)}{IPD} = \frac{\sin(90 + a_r)}{D_l}$$

$$\frac{IPD\ \cos(a_r)}{\sin(a_l - a_r)} = D_l \quad [Eqn.4]$$

Distance to the target is defined as the length between the vergence point of the eyes, and the centre point of the eyes. If a line is drawn between these points, the triangle would be cut in two. If the right triangle is inspected, the one drawn between the target, the centre position, and the left eye, the length of one of its sides is the distance to the target. As the other two side lengths are known (IPD/2 and D_l), and the angle between them is known ($90 - a_l$), the cosine rule can be used to find the final triangle length (Equation.5):

$$D_c{}^2 = D_l{}^2 + \left(\frac{IPD}{2}\right)^2 - 2 \cdot D_l \cdot \frac{IPD}{2} \cdot cos(90 - a_l)$$

$$D_c = \sqrt{D_l{}^2 + \frac{IPD^2}{4} - D_l \cdot IPD \cdot sin(a_l)} \quad [Eqn.5]$$

To implement this equation, both eyes must be programmed to track a target in order to measure the angle of each eye. The right eye will be following a colour target, whereas the left eye will be using normalised cross correlation (Equation. 6) to find what the right eye is pointed at, by matching a small window of pixels in the centre of the right eye across the entirety of the left FOV. Programming the eyes in this fashion makes for a robust tracking system, as the left eye is always tracking what the right eye is looking at, regardless of the target.

$$R(x,y) = \frac{\sum_{x',y'}(T'(x',y') \cdot I'(x+x',y+y'))}{\sqrt{\sum_{x',y'} T'(x',y')^2 \cdot \sum_{x',y'} I'(x+x',y+y')^2}} \quad [Eqn.6]$$

where T' and I' are template and image respectively, both normalised by size and by average intensity (see OpenCV Template Matching tutorial)

A tape measure was placed against the centre of the owl body and extended perpendicularly. The target was placed in front of the owl, in line with the tape measure and at the same height as the cameras. Starting from 130mm, the target was moved back to 430mm. The estimated distance from the camera vergence was recorded in 50mm increments. This process was repeated four times to reduce measurement error.

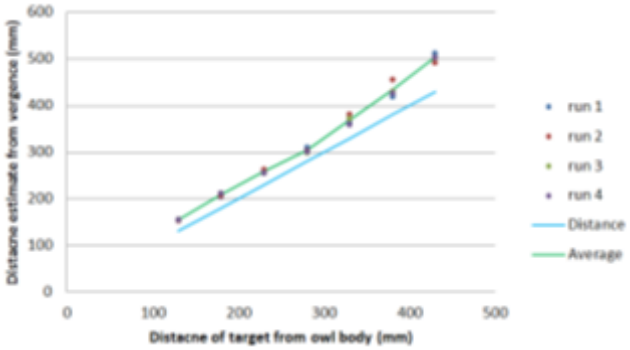

Figure 3: Distance estimation from initial vergence experiment

Figure 3 shows how the distance estimate compares to the true distance. The blue line is the ideal characteristic that the algorithm would achieve, and the green line is the average response of the distance estimate. Error between the two increases over distance, however expressed as a percentage the error is more consistent. This is evidence for the IPD measurement being incorrect, as this is the scale that the algorithm uses to define a millimetre. The average percentage error was calculated to be 14%, thus the IPD constant within the program was modified from 66.5mm to 58.3mm. With this correction, the experiment was repeated.

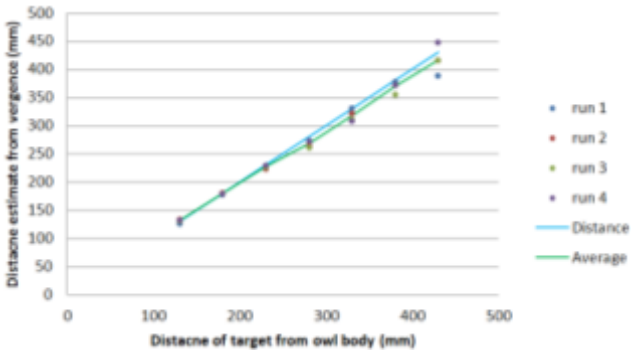

Fig.4: Distance estimation via vergence experimental results 2

Compared to the previous experiment, Fig.4 shows improvement, with an average percentage error of 1.96% below the true distance. For a demonstration of vergence tracking and distance estimation, follow this link: *https://youtu.be/KBRrzwK_Kjg*. This error was due to the cameras being mounted with the lens being at the centre of eye rotational axis, and not the silicon sensor.

Stereo calibration and disparity calculation

Although each camera has been calibrated for lens distortions and other intrinsic errors, they need to be calibrated together as a stereo pair, correcting for extrinsic distortions and establishing a common three-dimensional baseline. Again, OpenCV provides an example program which does this (stereo_calib.cpp) The software similarly uses an XML file to input parameters, this time requiring the file locations of any number of stereo pair images. A simple program was written to take images when a key was pressed, and save them as a pair of JPG files. Nineteen images pairs were taken, each containing the checkerboard calibration target at different positions and orientations. Fig.5 shows two example pairs used for the calibration.

Fig.5: Two stereo pairs used for stereo calibration

After processing the program stated that the RMS error was -0.606984, and the average epipolar error was -0.553882, which was acceptable. The distortion correction was applied to the calibration images and displayed (Fig.6), note that the green guides on the images align with the exact same areas on the checkerboard in both camera views, this indicates that the calibration worked as intended.

Fig.6: Two stereo pairs corrected for extrinsic distortions, showing epipolar lines

The compute function in StereoSGBM (in OpenCV stereo-match.cpp) was used to create the disparity map, which was then normalised so it could be viewed in a visualizer. Fig.7 shows the generated disparity map. For the relatively low resolution, the map looked surprisingly good, given no additional noise reduction. As expected, this is very error prone, but for a rough measure of depth, a disparity map may be satisfactory. The left side of the image has been seemingly removed. However, this is just a result of the reduced FOV when overlapping two cameras.

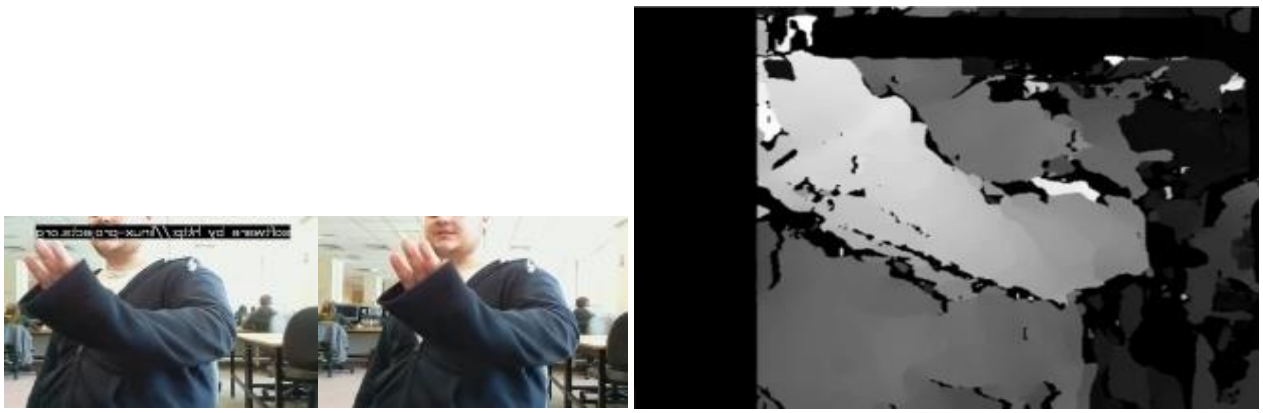

Figure 7: Stereo images and resulting disparity map

The brightness of each pixel represents the disparity of that physical object in both camera views, very similar to how a single target was tracked in the previous experiments. To see how

this disparity relates to depth, an experiment was set up. A tape measure was laid out perpendicularly to the robot, and a target placed at set distances.

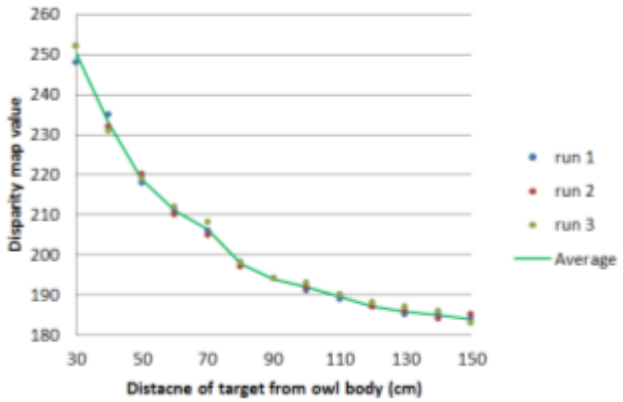

Figure 8: Brightness of a target vs its distance from the owl

The data in Fig.8 shows that the relationship between distance and disparity is non-linear. Disparity and distance are trigonometrically related via the tangent function. As the units are ambiguous a model will attempt to estimate the disparity for a given distance, as there exists experimental data to test against. Once a model has been calibrated, it can be reversed to calculate distance for a given disparity. Disparity values were inverted to start from 0 instead of 255, and an arctangent (Distance) function was plotted, see Fig.9.

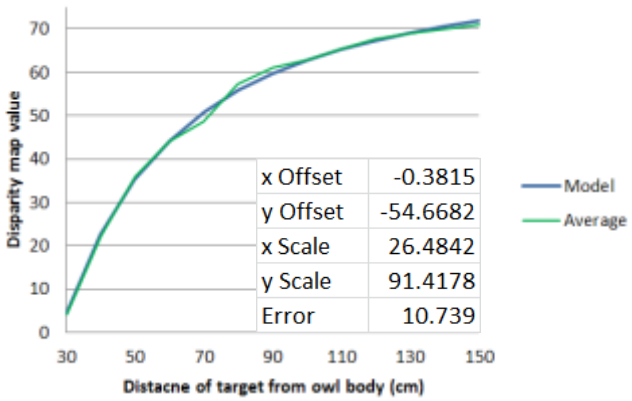

Figure 9: Model for estimating disparity for a given target

From the graph it is clear that the model fits closely with the experimental data. With this model calibrated, the final conversion between disparity and distance is as so:

$$Distance = 26.4842 \cdot \left(tan\left(\frac{Disparity + 54.6682}{91.4178} \right) + 0.3815 \right)$$

A demonstration on distance estimation via disparity can be found here:
https://youtu.be/KBRrzwK_Kjg?t=1m3s

Saccadic eye motion and Saliency

The human eye has a central region of high resolution, called the fovea. This is so that the body's limited resources are focused where they are needed, providing a high-fidelity view of what is being attended to, with a more general, lower resolution, view in the periphery. Saccadic eye motions make use of this small area, by rapidly moving the fovea over a set of salient targets, allowing the brain to build up a more detailed view of an object or scene much larger than this small area of high resolution. The motions are ballistic, and cannot be corrected for the duration of the movement. If the desired target moves as the eye is saccading, the eye will miss the target and a second saccade will be required to correct (Purves et al., 2001).

Saccades can occur both voluntarily and reflexively, demonstrating a combination of bottom-up and top-down processing. As reflexive eye saccades are based solely on the information contained within the scene, computer models can attempt to imitate how this is done in humans.

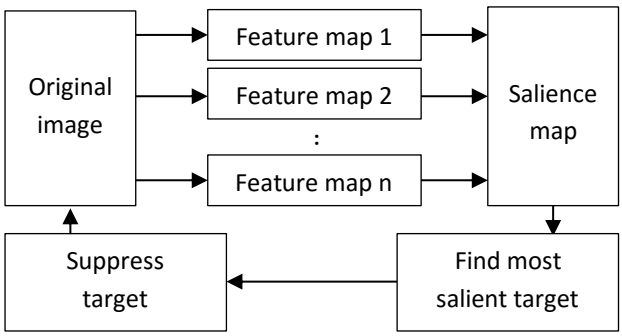

Figure 10: Salience based visual attention model (after Itti & Koch, 2000)

Itti & Koch (2000) proposed a model based on salience (Fig.10), where feature maps are created form the environment, and combined into what can be described as "interest map" with which a vision system can sequentially direct its attention. A feature map will highlight a single property about the scene, such as colour or orientation. Multiple feature maps means that the visual model will take into account more information about the visual stimulus. A saliency map is the linear combination of these feature maps, each weighted on their importance. Peaks in the salience map are targets that a human's visual system may find interesting and saccade to (Fig.11). One issue with this model is that its stable, and with a static image as stimuli, the model would saccade to the most salient target and not anywhere else. The model proposed by Itti solves this problem by deleting the region in the stimuli, where the model found the last salient target. This would make the second most salient target the next location to saccade, and so on. A limitation with this solution is that it assumes that salient targets are only looked at once.

A participant was given an image of a building in a free viewing condition, and their eye movements were recorded for 5 seconds. Highly salient points such as a window reflection are visited more than once, as regions of interest may be too detailed for a single glance to capture. Whatever the reason, simply preventing a visual attention system from visiting the same spot twice is not true to what is observed in humans.

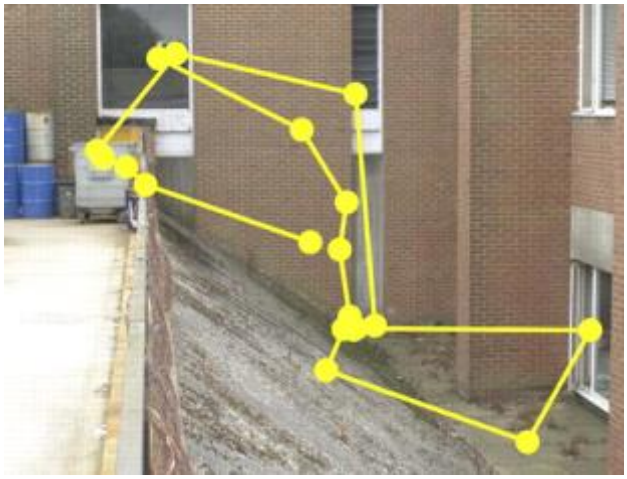

Figure 11: Saccadic movement from a single participant, observing the stimuli for 5 seconds. (source: Tatler et al. 2011)

A solution is to implement a familiarity map, where the locations of salient targets are added as faint blobs with a low opacity. Areas commonly looked at will get darker as many of these blobs overlap. This map could be seen as memory, where light regions indicate little knowledge of an area. If this is included as a feature map, highly salient targets will be supressed as they are observed. However, if a target is sufficiently interesting, this model allows for the visual attention system to re-visit a target. The visual attention system was developed on static images, allowing saccades over static scenes, so that the stimuli are more consistent and changes to the program can more easily be monitored. The maps that were decided upon were edges, colour, and detail. Colour is based on how saturated and bright the colours are in the scene, detail activates when many edges are detected in a small space, and edges highlight areas of contrast as its unlikely that smooth featureless areas will become saccade targets.

Colour Map

To evaluate the colour within the room, it would be a lot easier if the colour format of the video stream was encoded as Hue, Saturation and Value (HSV). This format arranges colour information is a convenient way, as the “strength” of the colour is contained in the saturation and value variables. For a colour to stand out in a scene, it has to be bright compared to the background. However a bright shade of white is not that striking compared to a bright green or other colour. Thus saturation is important. To create a map that rewards a combination of value and saturation, these variables for each pixel are multiplied together:

As hue is not important in this map, the spare channel in the matrix was used to store the product of saturation and value. Fig.13 shows the result of this filtering. The orange wall in the background and the computer monitors are quite colourful, which correspond to high values on the colour map.

Figure 13: Colour feature map, processed from the saturation and value of the scene

Detail Map

The detail map is to reward high concentrations of edges in small areas, such as a keyboard or text. To do this a high-frequency Canny edge detector created a binary view of the scene, populated with edges. This was morphologically filtered using the dilate and erode commands, this has the effect of filling in the gaps when edges are close to one another. With the view filled with blotches of white wherever there are high concentrations of edges, the image is blurred. Small groups of edges will blend into the background black, however larger groups will persist, and gain a radial gradient with a peak at the centre.

Figure 14: Detail feature map

The resulting feature map is shown in Figure 14. Note that the air vent, poster and signage appear as bright spots in the processed map. With this feature map incorporated into the attention system, highly detailed objects like these will have greater chance of being salient targets.

Edge Map

The salience map so far is a combination of colour and detail, and it can highlight low-resolution areas of interest. However, specifically where to look within these salient patches is not evaluated. Adding an edge map will give the attention system a preference for observing

contrasting areas. If a colourful and detailed sign was in frame for example, instead of simply looking at its centre, the edge map will reward looking at the text or symbol on the sign.

There is evidence that the human eye processes edges before the signal reaches the brain. Such processing is done by ganglion cells within the retina, and are the first cells to signal an action potential from the sensor cells to the brain (Nelson, 2010). Each cell responds to stimuli with their receptive fields, which is a small region of cone (colour) and rod (monochromatic) sensor cells. The receptive field is split into two sub-regions, centre and surround (Fig.15)

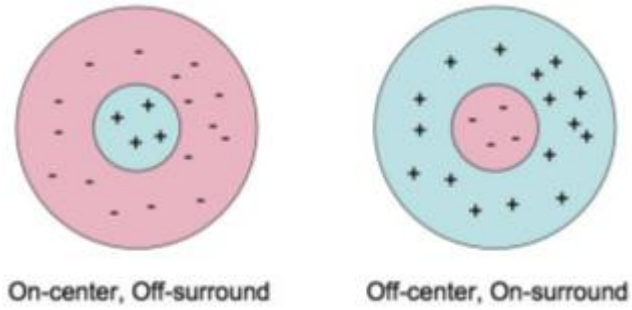

Figure 15: Centre surround structure of a ganglion cells receptive field, both on and off centre types (adapted from Marr, 1982)

These cells do not respond to featureless stimuli, as subtracting one sub region from the other results in zero, because the average activation of each will be the same. The result is approximated as a Difference of Gaussians (DoG), see Marr, 1982. However, something differing over receptive field, such as an edge, will elicit a response (Fig.16 shows an image and a DoG filtered result).

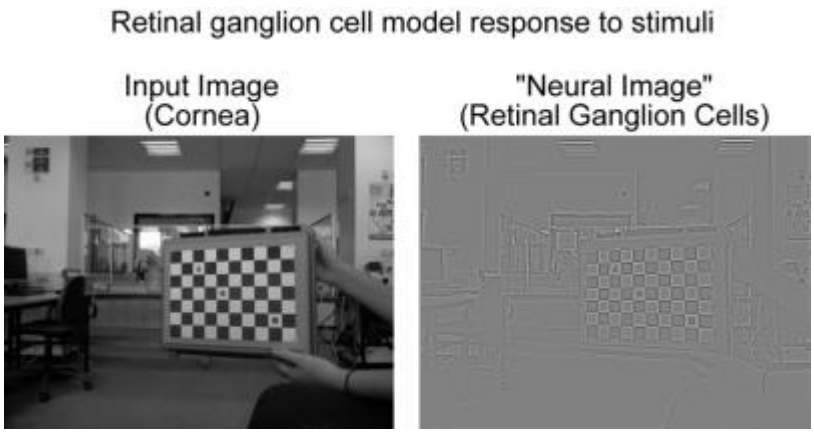

Fig.16: Computer model of ganglion response.

Such edge detection can be derived from the application of a Difference of Gaussians (DoG) filter over an image. See Marr (1982) for a discussion and Assirati et al. (2013) for an example.

Figure 17: Edge feature map based on DoG filter

This map (Fig.17) was deemed to highlight edges sufficiently, and be satisfactory for use in the salience map.

Local Map

The local map needs to appear like a spotlight, with a gradient of 255 at the current salient target, trailing off to 0 at the extremes. To achieve this, a large white filled circle is drawn where the previous salient target was. A strong Gaussian blur smooths this out into a spotlight with a single peak. (Fig.18)

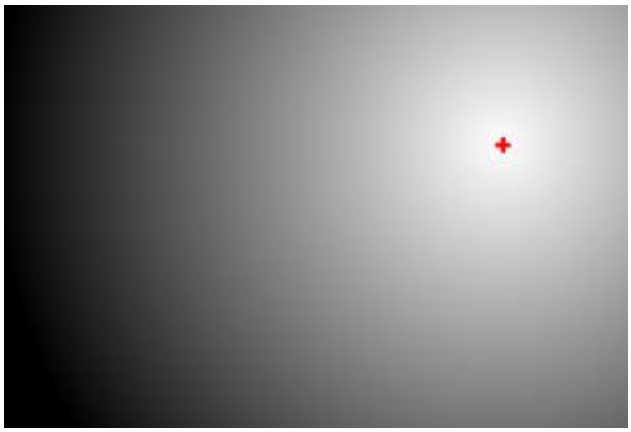

Figure 18: Local map for a target located at the red mark

Familiarity Map

This map (Fig.19) keeps track of the most visited areas of the attention system, darkening areas of the map that are observed. Over time, frequently observed regions will get darker, making the subsequent observation of these areas more unlikely. This has the effect of supressing salient targets, without preventing multiple observations of the same point.

Figure 19: Familiarity map of a target after 150 seconds of saccading. Note darker regions are not saccaded to, but gradually return to white and the Owl can become interested in them again.

Dimming a region will require opacity, which can be somewhat achieved with the "addWeighted" command. A blank white map is initialized. When a salient target is saccaded to, this map is duplicated. One version gets a solid black mark, and blurred, whereas the other version is unchanged. The final updated map is a weighted sum of the two versions, and the strength of the memory effect can be adjusted by changing these weights. This should appear as semi-transparent blurred circles plotting onto a white background, to create an accumulative history of what has been observed. The memory can have a variable persistence over time, forgetting the oldest saccades plotted on the global map.

Salience Map

With all the feature maps processed, they are linearly combined into a salience map using a weight. The familiarity map is different as it supresses salient targets, and a zero in this map must be a zero in the salience map. So in this case, this feature map is multiplied to the weighted sum of the other maps. This map is scaled and normalised to convert the greyscale salience map into a hue-scale visualisation, blue being the lowest salience, and red the highest:

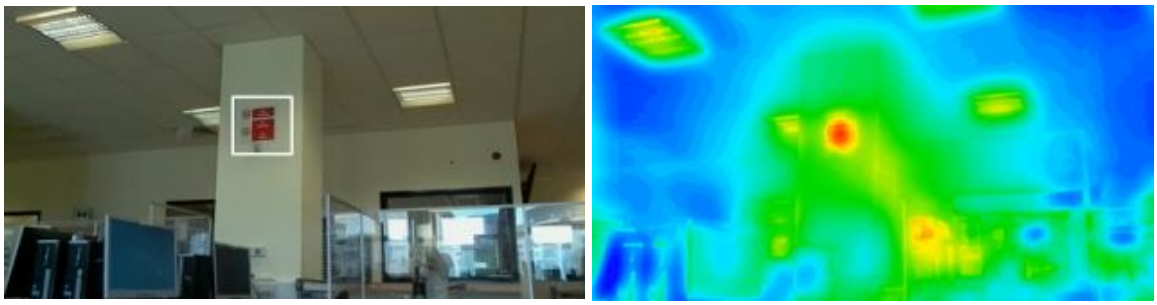

Figure 20: Salience map.

Interestingly the visual attention system finds a hazard sign to be the most salient target (Fig.20). By design these signs attempt to catch the attention of humans, so the fact the computer model also responded in the same way is a promising start.

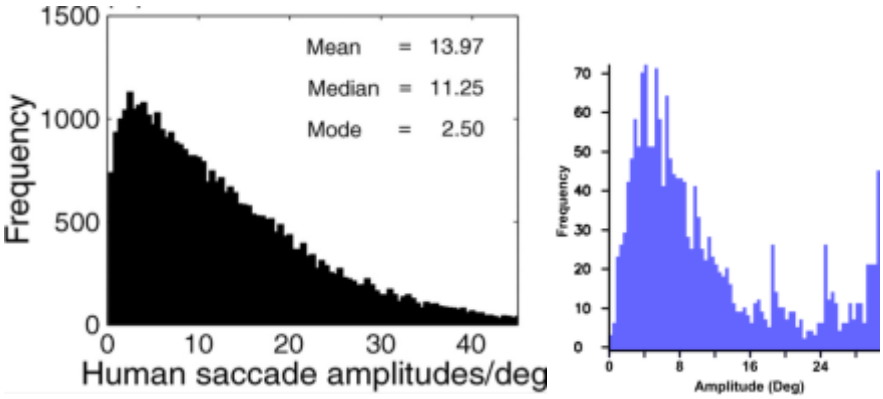

Figure 21: Comparison of saccading visual systems: (a) human (left), from Tatler et al., 2011 and (b) the Owl model (right)

An important characteristic about human saccades is a magnitude versus frequency plot, and is an easy test to apply to the synthetic attention system. This characteristic made adjusting the local map easy, as changing this weight affected the size of the saccades. After calibrating, this is how the human (Fig.21a) and the current visual model compare (Fig.21b). The graph has a clear peak of low magnitude saccades and a linear drop off in amplitude. This linear region in steeper for the owl compared a human, though this could be due to the restrictive FOV of the cameras compared to a human eye. There is a strange peak of high amplitude saccades at around 30 degrees of visual angle. Observing the video footage, the owl seems to catch its own eye-socket every now and then. Since the owl frame was bright red, this can trigger a salient target at the corner of the FOV, accumulating in a higher than average peak at this amplitude.

Over a large enough sample set, one would assume that there would be little relation between the size of a saccade and its frequency, when using the current proposed salience model for visual attention. This due to the current point of attention having no effect on the next (excluding the familiarity map which is a very local effect). Figure 21 shows that with human saccades, there is a high bias for looking at salient targets near the fovea. To implement this into the model, a spotlight-like feature map will be created to track the previous area of attention. This will have the effect of amplifying the salience of features near to the fovea of the attention model. The general shape of the curve is very similar to that of a human. Small motions are more likely than large ones, tiny motions however are less likely, and the frequency drop-off has a linear region. To test the similarity of the models saccade targets to a human's, experimental data of a human's response to stimuli is required, along with the original stimuli. Online research returned few examples, but the best that could be found were tests done on faces (Fig.22).

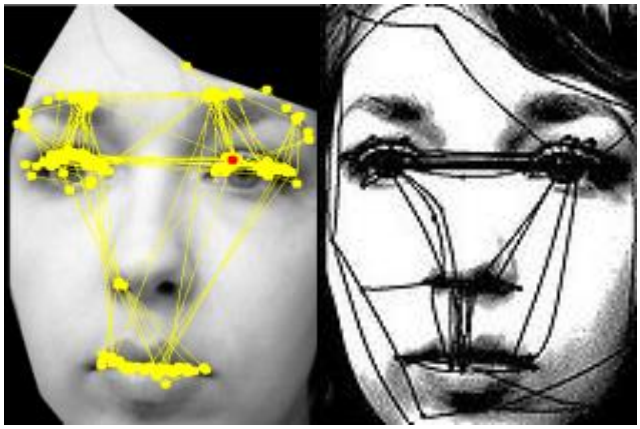

Fig.22: Comparison of saccade paths and targets between (a) the OWL visual attention system (left) and (b) a human (right) (source https://en.wikipedia.org/wiki/Saccade)

Owl robot deployment

Minimal modification is required to make the software ready for the owl, since the image processed is taken from the left camera rather than from a pre-loaded static image, and the most salient target coordinates are used to move the cameras and neck. To find the right corrective servo movement needed to centre the next salient target, the difference between the centre of the image and the new target is to be calculated. This is the trajectory of the next saccade. The differences are converted from pixels to degrees, and sent to a function that moves the servos. The right camera is moved so that it is always parallel to the left camera, using the lefts absolute position to generate a corrective trajectory. The neck loosely tries to keep the eyes at the centre of their horizontal movement range. The global position of the left eye is subtracted from the calibration point, which is the motor position for looking straight ahead relative to the owl head. A number is generated that indicates how off centre the eyes are.

The familiarity map will have to change as salient targets in view of the camera will not stay in the same position on the screen as the cameras move. A solution to this is to create a map that takes into account the motor positions. Assuming that the owl body is not moved, and assuming the environment is mostly static, salient targets will always be in the same physical location. To create this new global map, not much is changed apart from the larger size. New updates to the map are offset by the current servo positions to put them into a global or panoramic frame. The only change that was made was to create a “GlobalPos” variable that stored the offset to the salient target position. The weighted-add function was given a minor edit to make the opacity variable. This global position is based on the servo position, converted to pixel units. When this larger familiarity map is used for the salience map, a smaller window is cut from it, representing the current view of the cameras. The local map was changed so that the spotlight gradient was permanently centred on the screen, as the fovea of the camera doesn’t move relative to the camera. The robot could now freely view the environment, saccading once every 400-500ms. Again, the magnitude of each saccade was recorded, so that it the frequency of chosen eye movements could be measured against a human’s.

Follow this link for a demonstration of the owl saccading about its environment:
https://youtu.be/KBRrzwK_Kjg?t=4m25s

Conclusions

The OWL robot offers a simple TCP/IP interface for the exploration of stereo verged and static eye motions for distance estimation from a host computer. Software has been developed that demonstrates distance measurements and target tracking using the OpenCV computer vision library. A simple model of visual attention has been developed that applies saccadic eye control which is driven by bottom-up analysis of scene features including colour saturation, edge density and orientation. Motion and other feature maps can be added to enrich the model.

The Plymouth OWL robot offers an open programming interface to explore advanced topics in cognitive vision. It has been used at Plymouth to explore animal vision as a source of inspiration for robot vision systems in module AINT308.

Acknowledgements

We thank Bill Stephenson, Martin R Simpson and Clare Simpson (now at Babcock) from the School of Computing, Electronics & Mathematics, for their design skills in creating the OWL robot to be such an engaging robot for teaching robotics visual perception engineering at Plymouth.

References:

Abrams RA, Meyer DE and Kornblum S. (1989) Journal of Experimental Psychology: Human Perception and Performance. 1989, Vol. 15, No. 3, 529-543

Assirati L, Silva NR, Berton L, Lopes AA, Bruno OM (2013) Performing edge detection by Difference of Gaussians using q-Gaussian kernels. 2nd Int.Conf. on Mathematical Modeling in Physical Sciences 2013 IOP Publishing. Journal of Physics: Conference Series 490 (2014) 012020 doi:10.1088/1742-6596/490/1/012020

Itti L and Koch C (2000) A saliency-based search mechanism for overt and covert shifts of visual attention. Vision Research 40 1489–1506.

Marr D (1982) *Vision*. WH Freeman and Co., 1982.

Nelson R (2007) Visual Responses of Ganglion Cells. In *Webvision: The Organization of the Retina and Visual System* Kolb H, Fernandez E, Nelson R, editors. Salt Lake City (UT): University of Utah Health Sciences Center (1995-.).

Purves D, Augustine GJ, Fitzpatrick D, et al., editors (2001) *Neuroscience: Types of Eye Movements and Their Functions*. Sinauer Associates. Second edition 2001.

Shelton J and Kumar GP (2010) Comparison between Auditory and Visual Simple Reaction Times. Neuroscience & Medicine, 2010, 1, 30-32 doi:10.4236/nm.2010.11004

Tatler BW, Hayhoe MM, Land MF and Ballard DH (2011) Eye guidance in natural vision: Reinterpreting salience. Journal of Vision May 2011, Vol.11, 5. doi:10.1167/11.5.5

Wilson SJ, Glue P, Ball D and Nutt DJ (1993) Saccadic eye movement parameters in normal subjects. Electroencephalography and clinical Neurophysiology 86, 69-74.