

# Vision Processing on the Bunny Robot Humanoid Robot

Phil Culverhouse<sup>1</sup>, Samuel Martin<sup>1</sup>, Romain Talloneau<sup>1</sup>, Thomas Rodier<sup>1</sup>,  
Neill Hughes<sup>2</sup>, Peter Gibbons<sup>1</sup> and Guido Bugmann<sup>1</sup>

<sup>1</sup>School of Computing & Mathematics,  
<sup>2</sup>School of Engineering  
University of Plymouth, Plymouth PL4 8AA, UK.

<sup>1</sup>P.Culverhouse@plymouth.ac.uk

<sup>1</sup>G.Bugmann@plymouth.ac.uk

<sup>1</sup>P.Gibbons@plymouth.ac.uk

**Abstract**— The Bunny robot is a new humanoid robot platform for teaching, research and competition. It has been developed to provide a detailed case study for undergraduate and postgraduate teaching, but it is also a research tool and competition robot. It is based upon the Robotis humanoid servo skeleton to which a multi-processor card cage and skull have been added. Steerable cameras and FPGA video stream processing provide VGA resolution image capture and processing. This paper describes a range of developments designed to improve the robot's performance at football and athletics for RoboCup and FIRA competition. Vision processing additions are described in relation to their potential gains.

## I. INTRODUCTION

Visual perception is an important function in a mobile robot facilitating navigation, object avoidance and directed motion towards or away from visual stimuli. The concepts of active vision, of visual attention and gaze control, Ballard [1] argued, can speed up image processing for mobile robot navigation and scene analysis. This paper discusses aspects of active vision through hardware and software implementations on the Bunny Robot; and explores implementation issues of Simultaneous Location & Mapping (SLAM) and eye vergence control in a low-power budget humanoid robot.

The Bunny Robot has been developed as a platform for teaching robotics. It is a bipedal humanoid design with 28 DOF (Wolf et al. [2]), it weighs 2.25kg incl. batteries and is 60cm high including ears. It is based on the Bioloid humanoid servo skeleton from Robotis with Dynamixel AX12+ servos, but additionally has a central card cage that can hold five low power ARM-based processor circuit boards see Fig.1. The skull holds twin cameras, servos for head, eye and ear control, a 2-axis gyro and an FPGA for servo PWM, local processing and video stream pre-processing, see Fig.2 (left). The entire design has been modelled in Solidworks, supporting centre of gravity calculations, and animation to check mechanical function. The additional mechanical parts have all been rapid prototyped for ease of modification during development.

Up to five processors can be inserted into the card cage, these are currently StrongARM processors from Toradex, the XScale PXA270 (500MHz) and PXA320 integrated into

motherboards. The new OMAP 3530 processor from Texas Instruments is being integrated into new motherboard cards to reduce image sequence processing time and replace the XScale processors where required. These will slot into the body card cage to add to the heterogeneous cluster of processors running either winCE 5.0 or Linux. A backplane power supply provides efficient buck-regulation of 5v and 3.3v for processor, FPGA and head servos. Host USB and Ethernet connection is also provided to support RNDIS and remote desktop debugging and host communications as well as point to point IP-based robot processor communications (see Fig.2 schematic diagram). The FPGA supports special purpose computation for camera image-stream processing from the two steerable, 30 frames per second VGA-resolution, cameras to dedicated DMA-based direct camera interfaces on two of the embedded ARM processors.

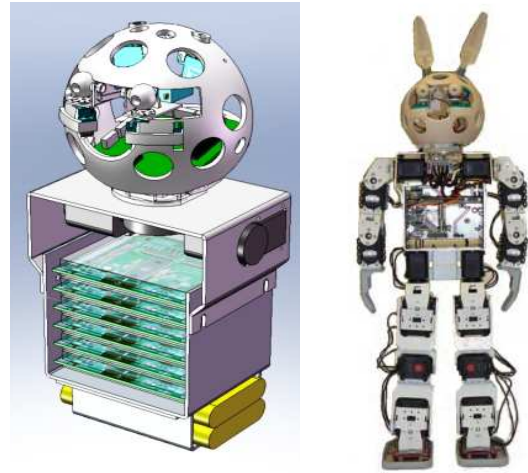


Fig. 1. (left) Solidworks model of the Bunny Robot torso showing processor locations, (right) photograph of Bunny robot. The head has twin cameras for eyes, an FPGA, gyro and temperature sensor. The body has a card cage that holds 5 processors for control and cognition, and battery pack

The robot was designed from the outset to compete in robot competitions, as it was recognised that this provides a strong motivation for students to learn and to contribute to the

continued development of the robot. It also ensures that the robot design never stagnates. This paper presents some recent developments on visual perception processing, focussed on playing robot football.

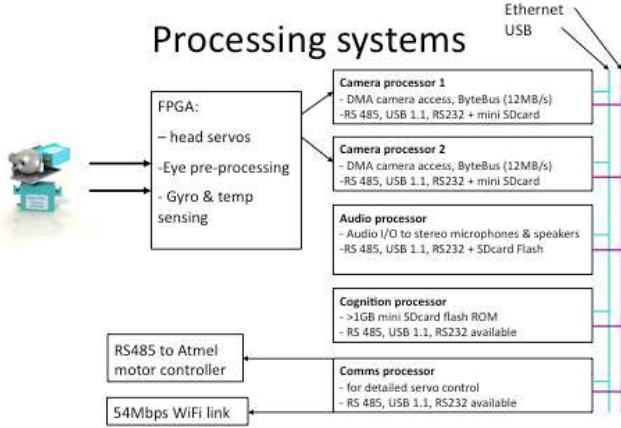


Fig. 2. Schematic diagram of processing function within the Bunny Robot

## II. VISION PROCESSING

The skull holds an Altera FPGA, type EP25C8T256 with approximately 25,000 macrocells, 609Kb RAM and 139 8x8 parallel multipliers making a 200MHz programmable special purpose processor. The device is used to provide real-time stream processing of the stereo camera video data at a combined data rate of 18.4MB per second (at 30 frames per second). The FPGA currently provides camera sync extraction, centre of mass colour blob tracking, and output streaming to two processors, with the potential to feed video data for SLAM analysis in one processor and for stereo disparity calculations in the other, for example. Three new functions have been added recently, i) sub-sampling and luminance/chrominance extraction of the video stream ii) eye vergence control, and iii) eye gaze stabilization. These are described below and facilitate ball tracking, navigation through SLAM, and object tracking whilst the robot is walking.

### A. The luminance sub-sampling process

This process transforms the video stream from the cameras (640x480 pixels, image format YUYV) to a luminance sub-sampled image stream (320x240 pixels, image format: Y) according to the flowchart (Fig. 3). This process supports eye vergence control at a resolution of 320 by 240 pixels.

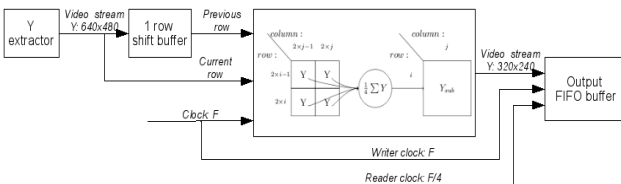


Fig. 3. Sub-sampling process flowchart

### B. The eye vergence controller

This design of an eye vergence controller was developed to provide depth perception to the Bunny robot. It is based on image patch correlation and triangulation calculus to recover the depth estimate and is implemented as hardware embedded in the FPGA.

The central ‘foveal’ patch from the left camera image, a sub window of 21 x21 pixels ( $L_{fov}$ ) taken from the centre of the camera image, is correlated with the whole right camera image (R) following eqn (2).

$$C_{fov} = \max \left( \frac{1}{n-1} \sum_{x,y} \frac{(f(x,y) - \bar{f})(t(x,y) - \bar{t})}{\sigma_f \sigma_t} \right), \quad Eqn. 1$$

where  $C_{fov}$  is the correlation coefficient,  $n$  the number of samples,  $f(x,y)$  the right image patch,  $t(x,y)$  the left ‘foveal’ template patch, and  $s_f, s_t$  standard deviations of intensity for  $f$  and  $t$  respectively.

But as the template  $t$  (left camera image fovea) and right image  $f$  are sampled at the same time, and both are approximately the same field of view, then the normalisation is not required. Hence eqn.1 reduces to a simple convolution operation, eqn2.

$$C_{fov} = \max \left( \sum_{x,y} (f(x,y))(t(x,y)) \right) \quad Eqn.2$$

The correlation is conducted between two image patches, one from the left camera fovea (central window) and one from the right camera ‘searching region’ (the slave camera) as shown in Fig. 4. The cameras are synchronised and so images are captured at the same instant. Then, the result of this correlation is used to control the slave eye along its two rotation axes by defining the disparities  $dx$  and  $dy$  and finding the location of the maximum of the correlation result (Fig. 4).

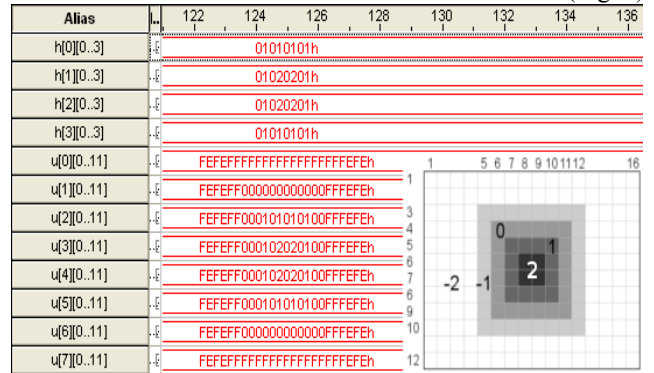


Fig. 4. Eye vergence flowchart

To run the correlation computation, it is necessary to store both input signals in on-board RAM before processing. The correlation is implemented as an auto-correlation operation of the left eye foveal region with whole field of the right eye. From the correlation result, it is simple to extract the disparity along each axis and control the slave eye-servos to make it gaze at the same object as the master one, in such way that the object is centred in both images.

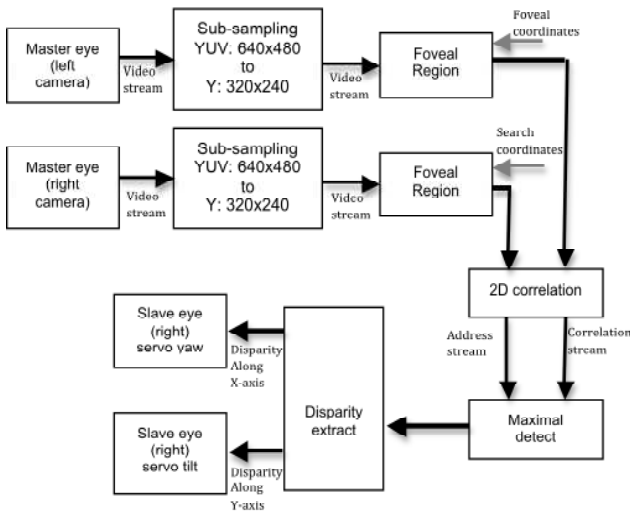


Fig. 5. Digital results of the correlation on camera signals from left camera  $h$  (source pattern to search) and right camera  $u$  (convolution result of match with target). Bottom right pattern is the correlation weighting function.

Due to some synchronisation issues inside the correlation operator (input signals on Fig. 5 and output on Fig. 6), the current implementation does not fully steer the slave eye, as it operates with some jitter. It should be noticed that, in this implementation, the master eye does not move.

Alias	-32	-16	0	16	32	48	64	80	96	112	128	144	160	176	192	208	224
convolution_result																	
u-patch_top-left_row									5								
u-patch_top-left_column																	
dx																	
dy																	
current weight																	
referenc weight																	
current row																	
current column																	
reference row																	
reference column																	

Fig. 6. FPGA signals, from the correlation result (the 3 top signals) to the generation of the disparity ones ( $dx$  and  $dy$ ). The 6 other signals are used to find the location of the maximum of the correlation result.

However this is easily corrected by balancing the vergence angle simultaneously to both cameras, in concert with neck rotation Vergence can be set by defining a toe-in of both cameras to verge to a near-field point upon start-up, given that all features within the football pitch will be closer than infinity. An alternative is to iteratively cause a toe-in by a coordinated movement of both the neck servos and the eye servos to ensure that the cameras form an appropriate triangle, based upon a desire to make the an angle to the attended object, driven via a look-up table based upon master eye angle (Fig. 7).

Fast vergence control will provide fast depth estimation to target at 30 frames per second with a one-frame delay in processing.

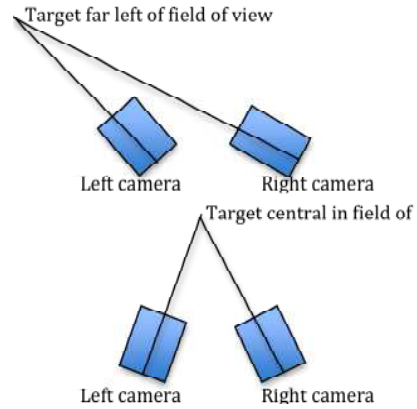


Fig. 7. Toe-in of steerable cameras optimises vergence and attention control

### C. The Gaze stabiliser

The twin cameras are mounted on X-Y servo controllers, allowing for directed gaze control. When mammals move their eyes, gaze is normally held to a point of attention. The same is required of the robot to maintain stability of navigation and other visual information whilst walking. The

robot has two integrated gyro units, the first a 2-axis device (MXD2020) is placed in the skull adjacent to the FPGA, the other a 5-axis HUVrobotics IMU is placed in the body card cage and is used to control gait. The 2-axis unit provides head pitch data at 8ms intervals. The Y-axis signal is processed and drives a servo to hold the eye gaze at a viewpoint, using Supertec Titch44 lightweight (4.5g) digital servos with an angular velocity of 60 deg/sec.

Gaze stabilisation is necessary to maximise the number of “target in field-of-view” image frames that can be extracted from the camera video stream whilst the humanoid robot is moving. Without gaze stabilisation it is common to only enable the camera stream processing when the robot is stationary.

The skull has a MEMS thermal gas bubble accelerometer (Memsic MXD2020e) that is designed to provide gaze stabilisation control, mimicking the vestibular ocular reflex (VOR) in primates. The inclination angle of the accelerometer

is given as  $\alpha_y = \sin^{-1} \frac{Ay}{g}$ , where  $Ay$  is the accelerometer output and  $g$  the acceleration due to gravity. The device can be treated as linear for inclinations between  $\pm 50^\circ$ . The servos have an angular velocity of  $60^\circ \text{ sec}^{-1}$  and are driven from a PWM source that requires a signal ranging from 1.5ms to 2.5ms for a  $\pm 90^\circ$  rotation. The user-set eye gaze y-axis can be modulated by the accelerometer output as described in Eqn.3. to achieve gaze stabilisation.

$$y_l = y_d + (k\alpha_y - g_{offset}), \quad \text{Eqn 3.}$$

where  $y_d$  is a externally set y-axis fixation,  $\alpha_y$  is the inclination angle from head gyro,  $g_{offset}$  to compensate for the gyro static pose component (the circuit board inclination when the skull is upright), and  $k$  a gain factor.  $K$  is set to 3. Updates are calculated every 8ms.

Experiments with stabilisation show that the mass and viscous components of the eye servos dampens the gyro position changes to achieve a smooth change in position, without PID control. Camera stabilisation for head movements above  $+50^\circ$  cannot be compensated as the accelerometer response flattens off. The head cannot move below  $-30^\circ$  and therefore does not require compensation for forward movement.

### III. IMAGE ANALYSIS

The Bunny Robot is able to stream video data from the twin cameras to selected processors in the on-board computer network. The design concept at present allows microprocessors to control camera source and to receive centre of mass calculations for target coloured objects from the FPGA stream processor. In the planning are more sophisticated analysis to allow the robot to self-localise and to recognise objects and scene scenarios. Two studies in preparation for this have been conducted on (a) localisation and mapping using a Beagleboard, and (b) stereo from verged images. The plan is to implement both these functions on new ARM-based processors that will be added to the Bunny Robot card cage, including the OMAP 3530.

#### A. Simultaneous Location and Mapping (SLAM)

A mobile robot has to navigate through its environment. The identification and tracking of landmarks in a visual scene is known as Simultaneous Location And Mapping (SLAM). Early work, for example by Smith et al. [3], was mostly based on active sensors such as laser range finders and sonar. These sensors usually provide a 2D map. More recently, for example Lacroix [4] and Davison et al. [5] have developed systems based on digital video camera images. Much work is currently focused on creating a reliable SLAM algorithm able to map large field of view precisely whilst accurately estimating the position of the camera at a computational cost.

A monocular SLAM system has been implemented for the Bunny Robot, using the Shi & Tomasi [6] corner detector to identify landmarks and image patch correlation to locate these in subsequent image frames using a  $21 \times 21$  patch size. An extended Kalman filter Orderud [7] is used to estimate 3D

camera pose based upon tracking up to 50 landmarks. Landmarks are forgotten if they fail to occur in a sequence of 10 image frames.

The work follows that of Davison et al. [8] and Civera [9] in using on monochromatic analysis of scenes to derive landmarks and in using image patch correlation for frame-to-frame patch matching.

Experiments were conducted with the SLAM algorithm to compare the speed of processing in Linux on a laptop and on a Beagleboard with an ARM-based OMAP 3530 processor. A dataset has been created in order to precisely measure the difference between the estimated positions and the real positions. This dataset is a simple translation of the camera in an indoor environment. Several public datasets available on the Internet e.g. the datasets of Civera et al. [10] have also been used in testing.

Table 1.  
Comparison of Processing Times for a Laptop and Beagleboard

Function call	Laptop	BeagleBoard
Shi-Tomasi detector	16ms	122.7ms
Matching by image patch	30ms - 300ms	500ms - 4s 462
Add new feature (1st)	293 $\mu$ s	1770 $\mu$ s
Add new feature (20 <sup>th</sup> )	9.236ms	167ms
Update (1st)	5ms	91ms
Update (20th)	16.5ms	349ms
Prediction (1st)	3ms	45ms
Prediction (20 <sup>th</sup> )	6ms	192ms
Measurement prediction	4-8ms	20-75ms
Display map	2ms	10ms
Total time (less verbose)	25s 790	5m 38s 29
Throughput	7.2 frames/sec	0.5 frames/sec

Table 1 shows the results of the comparisons of time to process 180 frames in a video sequence. The laptop is overall about 13 times faster than the OMAP processor on the 180-frame analysis, although parts of the processing show only a five-fold drop in performance. Overall throughput is 7.2 VGA-resolution frames per second for the laptop and 0.5 frames per second for the Beagleboard. The standard GNU gcc compiler was used together with openCV libraries. There has been no OMAP optimisation; hence timings can be expected to see a ten-fold improvement for floating point and convolution operations when the NEON co-processor is used effectively. Given that the OMAP 3530 is consuming only 300mw against the 15w of the laptop processor its performance is commendable.

There is scope for performance improvement. For example, the corner detection can be realised in the FPGA and the correlation could be realised in the DSP core of the OMAP 3530. More importantly, the functions ‘‘add new feature’’ and ‘‘update’’ could be modified to change only part of the covariance matrix of the system and thus avoiding the multiplications of large matrices.

The algorithm gives coherent results when all the parameters are well estimated, but a higher order SLAM algorithm needs to be created to add meaning to the corner features and to give a better estimate of a global map with a reduced computational time. Since the cameras in the Bunny Robot are relatively narrow angle (60 degree field of view) the information from each camera image is best used from this algorithm to estimate small regions of the environment and only predict the positions of the features in the next frames. The global mapping process still needs to be implemented.

### B. Stereo disparity from steerable cameras

Epipolar geometric correction (Hartley & Zisserman [11]) prior to disparity calculations is commonplace where twin cameras are not in perfect alignment. It is normal to verge cameras at infinity. Since the Bunny Robot possesses steerable cameras this geometric correction becomes pointless, as it has to be recalculated each time a camera is moved. An alternate correction is possible (Krotkov et al. [12]). Consider the case where two steerable cameras are verged, in both the X and Y planes, so that a small central portion of each image (a fovea) is overlapped with zero disparity (see Fig.7).

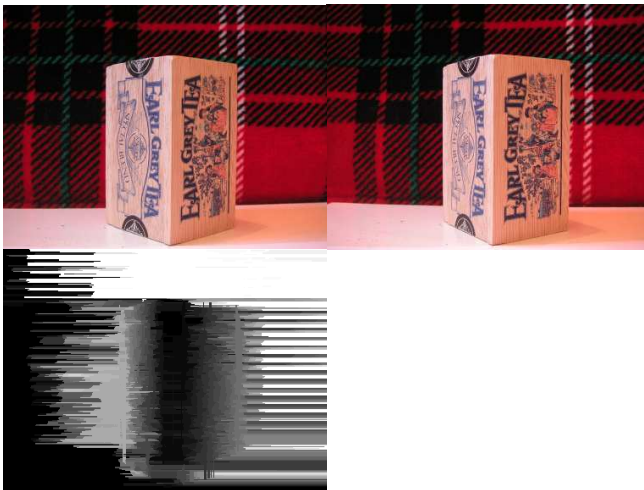


Fig. 7. (a) Images for the left camera (top-left) and (b) right camera (top-right). The cameras are verged onto the edge of the tea box. (c) The disparity map (bottom) shows the low disparity (dark colour) of the fixated edge from preliminary studies using the standard Birchfield and Tomasi (1999) algorithm.

In this verged case, the central (foveated) area of the stereo image has zero disparity, with a disparity increasing on each side of the foveated area (Fig. 7c). Disparity changes similarly for points nearer and further away from the cameras. The standard relation (for coplanar stereo cameras) between depth and disparity is valid no more. Standard disparity detection algorithms are also not able to operate optimally. One of the reasons is that corresponding points are generally not located on the same image row (see for example the different slope of the bottom of the tartan pattern in the left and right images). This results in highly noisy disparity maps (Fig. 7) when

calculated using standard stereo correspondence algorithms (Birchfield and Tomasi, [13]).

With verged vision, stereo disparity cannot inform on target distance. However, it contains information on the shape of the targeted object. This is of special value for the control of grasping. Mammalian vision is regularly processing verged stereo images (as shown recently by Bhattacharyya et al. [14]), but little is known on processing details, except that human gaze control is influenced by the muscular cost of accommodation of eyes, disparity and proximal cues [15]. New convolution-based algorithms will need to be developed to exploit the interesting properties of verged vision. Such algorithms also may have the potential to rapidly derive the relative distance of objects behind or in front of the verged target through crude disparity calculations. The distance to the target is then estimated through the vergence angle. Rapid calculations of relative depth can be helpful for judging the spatial relation between ball, goal and other players, and the 3D shape of the ball, in preparation for kicking.

### C. Conclusions

The basic Bunny Robot platform has been built and tested. New VGA-resolution image stream processing has been implemented using VHDL in an on-board FPGA that operates in real time at 30 frames per second. Furthermore scene analysis methods are being developed to add-value to the platform for navigation, object tracking and target distance estimation. These features are being implemented in 300mW ARM-based processors consuming less than 1.5w of power in total. These functions will be evaluated in competition in 2010. The Bunny robot platform design will be released as an open-source design in the near future.

### References

- [1] D. Ballard, "Active Vision", Artificial intelligence, 1991 vol 45, pp.57-86.
- [2] J. Wolf, A. Vicente, P. Gibbons, N. Gardiner, J. Tilbury, G. Bugmann and P. Culverhouse "BunnyBot: Humanoid Platform for Research and Teaching", *Proc. FIRA RoboWorld Congress 2009*, Incheon, Korea, August 16-20, 2009 (Springer Progress in Robotics: ISBN 978-3-642-03985-0), p. 25-33.
- [3] R. Smith, M. Self and P. Cheeseman, "Estimating uncertain spatial relationships in robotics", *Proc. IEEE Int. Conf. Robotics and Automation*, 1987 vol. 4, pp. 167-193.
- [4] S. Lacroix, "Cartographie et localisation simultanés par vision en robotique mobile", PhD Thesis, LAAS/CNRS, Toulouse 2006.
- [5] A. Davison, I. Reid, N. Molton and O. Stasse, "Monoslam: Real-time single camera slam", *Pattern Analysis and machine Intelligence, IEEE Transaction*, 2007 vol. 29, no. 6, pp. 1052-1067.
- [6] J. Shi and C. Tomasi, "Good features to track", *Proc. CVPR'94, Computer Vision and Pattern Recognition, IEEE Computer Society Conference*, 1994 pp. 593-600.
- [7] F. Orderud, "Comparison of Kalman Filter Estimation Approaches for State Space Models with Nonlinear Measurements". SIMS2005 - Scandinavian Conference on Simulation and Modeling, Trondheim, Norway, 2005 October 13-14.
- [8] J. Civera, A. Davison, J.M. Montiel, "Inverse depth parametrization for monocular slam", *Robotics, IEEE Transaction*, 2008 vol. 24, no. 5, pp. 932-945.
- [9] J. Civera, A. Davison, J.M. Montiel, "Slam using monocular vision dataset" 2006. Accessed online 23 July 2009. URL: <http://www.robots.ox.ac.uk/~SSS06/Website/index.html>

- [10] R.I. Hartley and A. Zisserman, "Multiple View Geometry" in *Computer Vision*, second ed., Cambridge University Press, 2004. ISBN: 0521540518.
- [11] R.P. Horaud and O. Monga, *Vision par ordinateur: outils fondamentaux*. Editions Hermès, Paris. 1995.
- [12] E. Krotkov, K. Henriksen and R. Kories, "Stereo Ranging with Verging Cameras," IEEE Transactions on Pattern Analysis and Machine Intelligence, 1990 vol. 12, no. 12, pp. 1200-1205.
- [13] S. Birchfield and C. Tomasi, "Depth Discontinuities by Pixel-to-Pixel Stereo", Int. Journal of Computer Vision, 1999, vol 35(3), pp. 269-293.
- [14] R. Bhattacharyya, S. Musallam S, and R.A. Andersen, "Parietal Reach Region Encodes Reach Depth Using Retinal Disparity and Vergence Angle Signals", J. Neurophysiol 2009 vol 102, pp. 805-816.
- [15] R.V. North, D.B. Henson and T.J. Smith, "Influence of proximal, accommodative and disparity stimuli upon the vergence system", *Ophthalmic and Physiological Optics*, 1993, vol 13(3) pp. 239-243.

#### ACKNOWLEDGMENT

The authors would like to thank Ian Philips of ARM Holdings Ltd for advice on ARM processor optimisation issues.