Gesture Recognition using High Resolution Stereo

Kasun Maldeni and Lochana Wijesundera and John Morris and Khurram Jawed
Department of Electrical and Computer Engineering
The University of Auckland, Auckland, New Zealand

***Abstract*—With high image resolution and depth resolution, it is expected that subtle differences in gestures can be detected and recognized and used for a wide variety of communication and control applications. As a first step towards this goal, verging axis stereovision was used to detect, segment and classify fingertips in high resolution images in real time.**
Firstly, binocular image points were extracted from the disparity map, smoothed and depth contours drawn. When a contour area increased by less than 1% when moving one disparity step from foreground to background, the foreground contour was taken as the hand outline. Fingertips were detected by drawing the convex hull around the hand and identifying the contour defects between fingers. The position of the finger and ratio of the lengths of its sides was used to classify fingers. This enabled us to recognize and classify the fingers of the hand as it was presented in its 'training' pose from which fingertip motion can be tracked and gestures understood.

I. Introduction

Gestures are an important component of our communication systems and are widely used. They augment normal speech, ' allow communication at a distance or in noisy environments and may be relatively immune to interference from other audio and visual sources. Computer recognition of gestures opens up many new applications from contactless control of computers and other devices, communication for the disabled to 'immersive' games and many more.

Most gestures involve movement in three dimensions and occlusions - for example, fingers crossed or turned in to form a fist - so that conventional 2D cameras have difficulty capturing enough information to interpret the full range of possible gestures - even if they can successfully separate a hand (or if we interpret 'gesture' as any movement which communicates something - limb or body) from its - possibly dynamic - background. Here, we used a high speed high resolution stereo system to capture '3D movies' of hands, isolate and identify fingers, enabling them to be tracked through rapid complex movements. This is a first, vital component of a reliable gesture recognition system: identifying and tracking fingertips should allow recognition of most gestures.

A. Related work

Most extant finger tracking studies have been based on 2D images and generally make assumptions about what the camera sees, *e.g.* one finger only [1] or the finger is at the highest point [2]. Skin colour based location and segmentation of hands has been used in several studies [3], [4], [5], but this depends on the lighting conditions and the absence of gloves, bandages, *etc.* Edge curvature has been used to generate finger locations [6]. Oka *et al.* have tracked multiple fingers from frame to frame using a Kalman filter [7] but they were limited to simple gestures on a 2D surface. In more recent work, Xing *et al.* used binocular stereo but also limited themselves to recognizing actions on a surface [3].

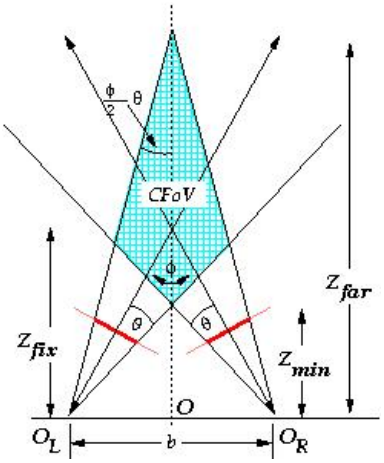

Fig. 1. Stereo converging axis configuration: the optical axes are verged to meet at a *fixation point* at a distance, Z_{fix}

B. Stereo Vision System

We used a high resolution stereo system developed at The University of Auckland which implements Gimel'farb's Symmetric Dynamic Programming Stereo (SDPS) algorithm [8] in an FPGA [9]. This system can stream left and right rectified images, disparity and occlusion maps into a host computer at 30 frames per second with an image resolution of 768×1024 pixels and 128 disparity levels giving depth resolution of $\sim 1\%$ over a wide area. Here we assumed hand gestures within a ~ 0.5 m cube would be sufficient *i.e.* no large arm movements. A verging axis configuration has better depth resolution [10] and we chose a vergence angle, $\phi = 5°$ - see Table I-B for the parameters of the system.

The system was configured so that we could obtain a depth resolution of better than our target ± 5 mm over a $0.34 \times 0.5 \times 0.4m$ region. The target depth resolution was easily achieved with the verging axis configuration (even at a low vergence angle) whereas a canonical (parallel axis) configuration could not produce this depth resolution from 1M pixel images.

TABLE I
CONFIGURATION PARAMETERS

Symbol	Description	Value
f	Camera focal length	9 mm
θ	Half angle camera field of view	15°
ϕ	Vergence angle between two optical axes	5°
b	Baseline: distance between optical axes	95 mm
d_{max}	Maximum disparity	127
p	Pixel width	4.7μ
$z(d_{max})$	Distance to closest point at which depth data can be obtained	652 mm
a	Extent: max width of object at d_{max}	340mm

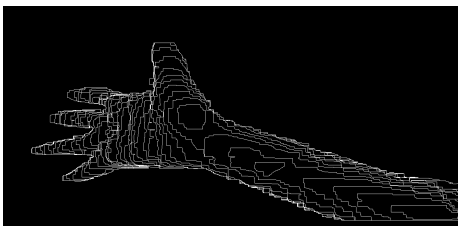

Fig. 2. Segmented and contoured hand - each contour represents $\sim$ 2 mm illustrating the high resolution capabilities of our stereo system.

II. System Operation

A. Calibration

The system was first calibrated to determine the camera intrinsic and extrinsic paramaters [11]. A rectification table was then constructed: the FPGA circuitry uses this table to remove lens distortion from the images and align them so that pairs of corresponding points are in corresponding scan lines. We used Bouget's technique [12] to rectify images without losing the enhanced depth resolution of a verging axis system [10].

New FPGA 'circuits' are built (including the rectification and SDPS correspondence circuits) and downloaded to the FPGA to which the cameras are attached: Jawed *et al.* provide a detailed description [9].

With Bouget's technique, we can obtain 3D point clouds in real world coordinates using the familiar relation between depth, z, and disparity, d: $z = \frac{fb}{pd}$.

The FPGA also sends to the host an occlusion map showing monocular regions - points visible by one camera only.

B. Hand calibration

We assumed that a new user would 'calibrate' the system by slowly moving a hand with the fingers spread out in front of the system to obtain some 'clean' images. Capturing images and depth maps at 30 fps, this training is fast and minimally invasive. We expect that we may be able to drop this set up exercise later - allowing the user to introduce the gesturing hand in any way at any time.

III. Finger Location and Classification

A. Hand segmentation

Using the algorithm of Figure III-A, hand segmentation is straightforward and robust.

The occlusion map selects binocularly visible points in the disparity map. Contours at each disparity level were then generated one at a time - starting with the closest (highest disparity) - using OpenCV's border following algorithm [11]: the map was binarized at each level, eroded and then dilated to remove noisy patches. Contours were smoothed using a polygon approximation from OpenCV. Figure 2 shows a set of contours.

The area of the contour at disparity, d, was compared with that of the contour at $d+1$. If the increase in area was less than 1%, this contour outlining the hand was extracted and used for fingertip detection. If not, the next lower disparity contour is tested.

```
Input: disparity map, threshhold
Select binocular points in disparity map
for d in 127 down to 0
  binarize disparity map at d
  dilate
  erode
  generate contour[d]
  a[d] = area of contour[d]
  if d < 127
    if (a[d]-a[d+1])/a[d] < threshhold
      return contour[d+1]
end for
```

captionContour selection algorithm

B. Fingertip detection

Two algorithms for fingertip detection were trialled. The first finds points of curvature by looking at gradient changes along the hand. The second finds convexity defects using OpenCV's `cvConvexityDefects` [11]. Manresa *et al.* also used convexity defects on hands identified by skin colour but did not attempt to classify fingers [5].

C. Points of curvature

The gradient at each point along the contour was calculated. Changes in gradient sign are points of curvature - circled in Figure 3.

This algorithm locates fingertips but finds additional points of curvature along the arm. The fingertips are spaced further apart than the arm's points of curvature which can be removed because they are too closely spaced. We also averaged the gradients along the hand to reduce the number of points of

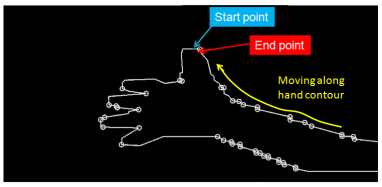

Fig. 3. Fingertip detection using points of curvature

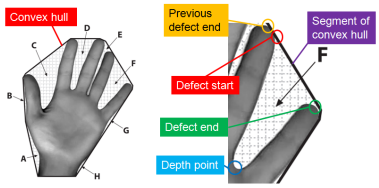

Fig. 4. Fingertip detection using convexity defects [11]

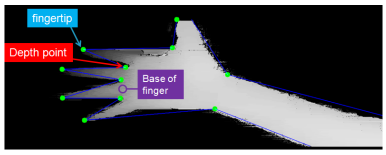

Fig. 5. Fingertips located using convexity defects on the disparity map

curvature found. This removed several points along the arm but also removed some of the fingertips. A more robust method to locate fingertips was required.

D. Convexity defects

Convexity defects are found by drawing the convex hull around the contour of the hand enclosing all points of the hand like a rubber band [11]. Regions where the contour departs from the convex hull are convexity defects. A defect is characterized by start, end and depth points. A set of defects (A to H) are shown in Figure 4:
The fingertip is the midpoint of the previous defect end and the defect start. Similarly, the base of a finger is the midpoint of two adjacent depth points - see Figure 4.
Convexity defects successfully found fingertips when disparity maps have minimal streaking artifacts.

E. Identifying fingers

1) Lengths of finger sides: The lengths of the sides of a finger were used to classify fingers. The two sides of each finger are labelled 'A' and 'B' - see Figure 6.
2) Angle about fingertip: The angle about the fingertip in Figure 7 is different for each finger but will always be acute. The arm's angle is larger than any of the fingertip angles. Restricting fingers to acute angles removes unwanted defects along the arm.

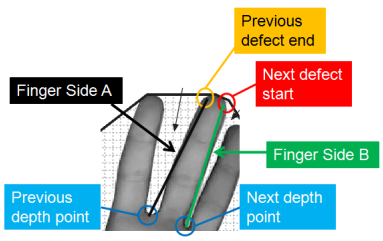

Fig. 6. Side lengths of a finger

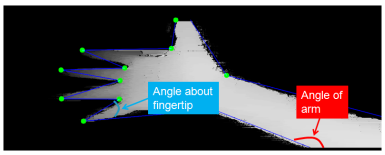

Fig. 7. Angles about fingertip and arm

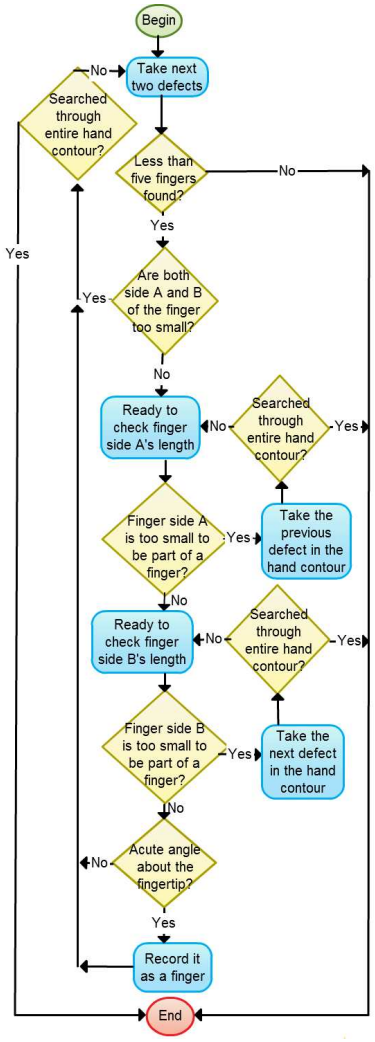

Fig. 8. Method for finding fingers

F. Method for finding fingers

Convexity defects may identify noise in the disparity map and parts of the arm as fingers. Defects due to noise have smaller lengths than the lengths of finger sides in Figure 6 and defects along the arm have obtuse angles. The procedure for removing unwanted defects and finding correct fingers is shown in Figure 8. It requires two defects to be selected, one for each side of the finger. If a finger side is too small, it looks for an adjacent defect. If finger side A is too small but side B is not, it means that side B is probably one side of a finger and side A is probably noise. Therefore, the previous defect in the array is

TABLE II
PARAMETERS FOR CLASSIFYING FINGERS

	Relative lengths of finger sides		Angle about the fingertip	
Fingers	**Mean**	σ	**Mean**	σ
Thumb	2.6	0.57	34°	9.7
Index	1.9	0.40	14°	5.6
Middle	1.1	0.19	11°	4.7
Ring	1.1	0.09	17°	3.2
Pinky	3.8	0.58	33°	11

checked for side A until it finds one of appropriate length or reaches the end of the defect list. Similarly, if finger side B is too small then the next defect is checked for side B until it finds one of appropriate length or reaches the end of the list. The procedure ends either when five fingers are found or the entire contour has been searched.

G. Finger classification

We first examined separate fingers, in order to make our classification robust to

- presentation angle of the training hand
- side presented (palm or back of hand) and
- left or right hand.

1) Parameters for classifying fingers: The following parameters were considered for classifying fingers - see Figure 6 and Figure 7:

- Relative lengths of the sides of a finger,
- Angle about fingertip and
- Average length of the two sides of a finger.

We found that the average length of the two sides of a finger was not a useful discriminant. Only the first two parameters had potential for classifying fingers.

2) Results: The relative lengths of the sides of a finger and the angle about the fingertip were estimated by running through 106 sets of images (left, right, disparity map and occlusion map) with low noise or steaking. Results are summarized in Table II. Angles about the fingertip had very large variations, especially the thumb and pinky. We concluded that fingertip angles will not accurately classify fingers but could be used to assist in verification. For instance, the angles about the thumb and pinky were mostly larger than other fingers.

Table II shows that the relative lengths of the sides of a finger have comparatively lower standard deviations. Figure 9 shows the majority of the relative lengths of the sides of a finger for the pinky are greater than for any other finger, followed by the thumb. It also shows that the middle and ring finger have much lower values than any of the other fingers. We concluded that the relative lengths of the sides of a finger is the best classifier of fingers.

H. Method for classifying fingers

When fingertip detection is completed, a list of fingers is found with a maximum of five entries: we now want to classify these fingers so that they can be individually tracked and used in gesture recognition: Figure 10 shows the algorithm used.

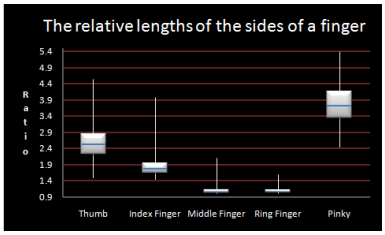

Fig. 9. Relative lengths of finger sides

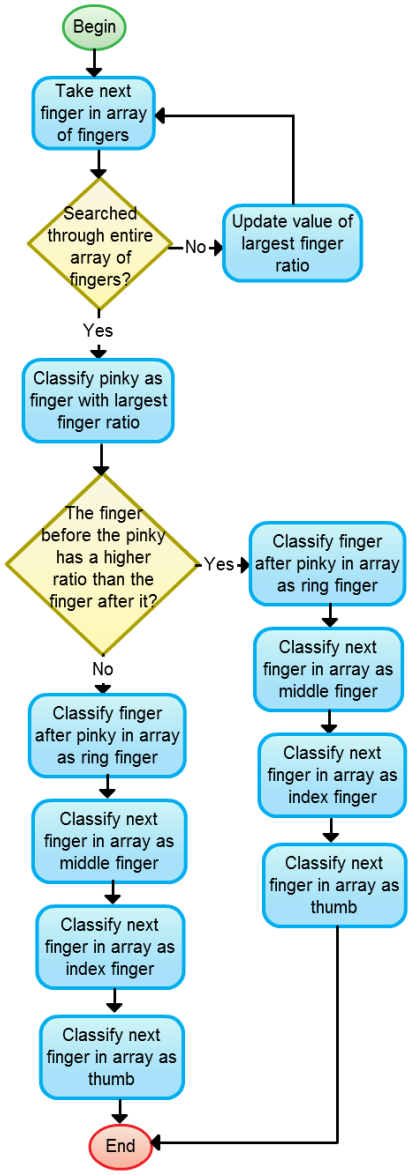

Fig. 10. Method for classifying fingers

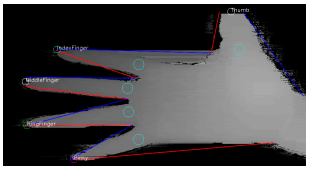

Fig. 11. All fingers classified correctly

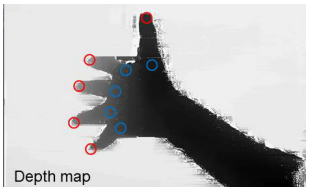

Fig. 12. *Depth map showing fingertips and finger bases*

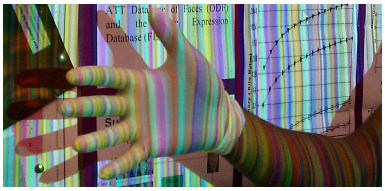

Fig. 13. Projected pattern of random width lines

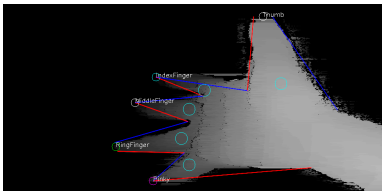

Fig. 14. Convexity defects finds bent fingers in disparity map

Noting that the finger list derived from the contours could start at any finger and run in either direction. The list is first searched for the finger with the highest relative length ratio. This finger is the pinky since Figure 9 shows the pinky consistently has the highest ratio. The ratios for fingers adjacent to the pinky are then compared. These are the thumb and ring finger. The ring finger ratio is almost always less than the thumb's, so the scan direction is easily established. The remaining fingers are trivially classified with the length ratios used to highlight outliers. All fingers are classified correctly now - see example in Figure 11. Since we acquire images at a fast rate, images which fail to produce consistent labelling can be simply rejected.

IV. Actual finger dimensions

The stereo system is calibrated so that we can convert a disparity map to a depth map with real world (X,Y,Z) coordinates for each pixel [11], which allow actual finger dimensions to be calculated. Actual finger dimensions can identify hand poses, such as the one in Figure 12, where the palm is not perpendicular to the system axis or the fingers are bent forward or back.

A. Limitations

1) Clear view of hand: The system must have an uninterrupted view of the training hand. The high frame rate removes this problem: the hand moves slowly and a suitable image set is quickly acquired.

2) Homogeneous skin texture: Hands tend to present mainly homogeneous, textureless regions. To overcome this, we projected a pattern of random width lines onto the hand - see Figure 13 - to reduce matching ambiguities. This improved the quality of depth maps significantly. All the images used in this work were taken with a monochrome camera so the colour pattern used here could easily be replaced with an 'invisible' near IR pattern. Conveniently, readily available (and cheap) Kinects emit just such a near IR pattern!

B. Fingertip detection

1) Fingers bent too far back: When fingers are angled away from the camera, the fingers can still be seen and classified correctly - see Figure 14. If they are bent any further than that, the index finger is no longer detected as a point in the convex hull. This puts a (not very significant) constraint on the angle at which the training hand must be presented.

2) Additional fingers from noise: The noise in some disparity maps generates a phantom finger in Figure 15. Such six (or more) finger hands can be rejected in training.

3) Streaking artefacts in the hand: Streaking is a well-known artefact of the dynamic programming correspondence algorithms, because they penalize disparity changes. Our high resolution disparity maps (depth resolution < 5 mm) show multiple contours over the full hand - for example, see Figure 2 - so, even when the disparity map is apparently corrupted by streaks, fingertips can often be detected and classified as shown in Figure 16.

Current work in our laboratory has found ways to improve this,

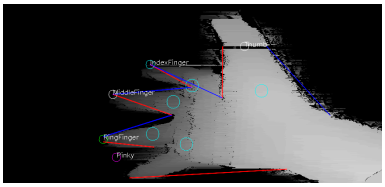

Fig. 15. *Noise detected as a pinky in the disparity map*

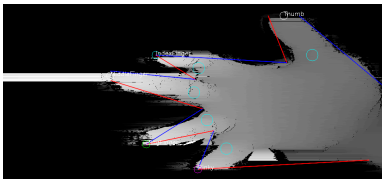

Fig. 16. Fingertips detected in disparity map containing streaking artefacts

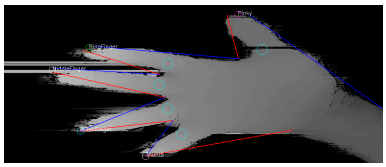

Fig. 17. Fingertips classified incorrectly by inverting fingers

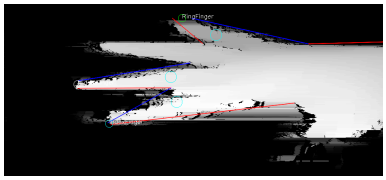

Fig. 18. *Some fingertips classified correctly when hand flips*

e.g. by using the Salmon algorithm [13], but not eliminate it. Comparison of the disparity map with the original images also shows promise in avoiding this problem: the key problem is the loss of the edge of the hand which distorts the derived contour, but we have *three* images in which to locate the correct edge - the original left and right images *and* the disparity map.

C. Finger classification

We searched the hand contour until five fingers were found or the entire contour had been searched. This is robust since less than five fingers can be outstretched and fingertips are still detected. Finger classification, however, relies on detection of all five fingertips. If less than five are detected, they may be classified incorrectly. This limits the hand poses in which fingers can be classified.

1) Relative lengths of the sides of a finger: Comparing the relative lengths of the sides of a finger, on rare occasions, incorrectly classify fingers as shown in Figure 17. The relative lengths of the sides of a finger are larger for the thumb than the pinky in this disparity map. This will cause the other fingers to classify incorrectly too.

2) Classification when hand flips: Figure 18 shows the hand flipped over so the back of the hand is facing the camera. The fingertips were detected are classified correctly. This further demonstates that relative lengths of the finger sides forms the basis for a robust classifier.

V. Conclusions

Our high resolution stereo system was able to identify a training hand, segment the hand in the image and identify and label each finger to assist in the next stage of gesture recognition. The 3D data trivially handles situations in which the finger lengths are distorted (in projection to the camera image planes) by bending them away or altering the hand pose. Our approach is robust to a degree of noise and streaking artefacts. When the classification fails, the system is capturing new images at 30 fps, so a failed image is simply rejected. To ensure that subsequent images do not suffer from the same problem (*e.g.* a background region which happens to look like skin), it is sufficient for the user to move their hand slightly in the training period.

The next stage in this work is to use a Kalman or particle filter to track individual fingertips through images captured every 30 ms - sufficiently fast that fingers do not move far in each frame. Here, the system will start analyzing each frame with a predicted position for each fingertip and will start with an expected disparity map against which the actual one can be compared. This will enable the system to handle complete occlusions of individual fingers (folded in, pointing away, one hand occluding another *etc.*) and build a set of fingertip tracks as input to the gesture recognition module. This study has shown that we can acquire very precise 3D views of a hand (as in Figure 2) and are thus able to distinguish subtle variations of gestures.

References

[1] H. Ying, J. Song, X. Ren, and W. Wang, "Fingertip detection and tracking using 2d and 3d information," in *7th World Congress Intelligent Control and Automation (WCICA 2008)*. IEEE, 2008, pp. 1149–1152.

[2] C. Hsieh, M. Tsai, and M. Su, "A fingertip extraction method and its application to handwritten alphanumeric characters recognition," in *IEEE Intl Conf Signal Image Technology and Internet Based Systems, SITIS08*. IEEE, 2008, pp. 293–300.

[3] J. Xing, W. Wang, W. Zhao, and J. Huang, "A novel multi-touch human-computer-interface based on binocular stereo vision," in *Proceedings of the 2009 International Symposium on Intelligent Ubiquitous Computing and Education*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 319–323. [Online]. Available: http://dl.acm.org/citation.cfm?id=1588297.1588984

[4] B. Lee and J. Chun, "Interactive manipulation of augmented objects in marker-less ar using vision-based hand interaction." in *ITNG'10*. IEEE, 2010, pp. 398–403.

[5] C. Manresa, J. Varona, R. Mas, and F. J. Perales, "Real –time hand tracking and gesture recognition for human-computer interaction," *Electronic Letters on Computer Vision and Image Analysis*, vol. 5, no. 3, 2005. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.102.1207; http://dmi.uib.es/ ugiv/papers/ELCVIAManresa.pdf

[6] B. Hamde, "Hands geometry as a human biometric: I know who you are from your hands!!" Department of Electrical and Computer Engineering, University of Auckland, Auckland, New Zealand, 2009.

[7] K. Oka, Y. Sato, and H. Koike, *Real-time tracking of multiple fingertips and gesture recognition for augmented desk interface systems*. IEEE Computer Society Washington, DC, USA, 2002, p. 429. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/AFGR.2002.1004191

[8] G. L. Gimel'farb, "Probabilistic regularisation and symmetry in binocular dynamic programming stereo," *Pattern Recognition Letters*, vol. 23, no. 4, pp. 431–442, 2002.

[9] K. Jawed, J. Morris, T. Khan, and G. Gimel'farb, "Real time rectification for stereo correspondence," in *7th IEEE/IFIP Intl Conf on Embedded and Ubiquitous Computing (EUC-09)*, J. Xue and J. Ma, Eds. IEEE CS Press, 2009, pp. 277–284.

[10] K. Jawed and J. Morris, "Verging axis stereophotogrammetry," in *Proc PSIVT'2011*. Springer-Verlag LNCS, 2011.

[11] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, Inc., 2008.

[12] J.-Y. Bouguet, "Camera calibration toolbox for Matlab," www.vision.caltech.edu/bouguetj/calib_doc, 1999.

[13] T. Khan, J. Morris, and K. Jawed, "Intelligent vision for mobile agents - contour maps in real time," in *ICAART 2010 - Proceedings of the International Conference on Agents and Artificial Intelligence, Volume 1 - Artificial Intelligence, Valencia, Spain, January 22-24, 2010*, J. Filipe, A. L. N. Fred, and B. Sharp, Eds. INSTICC Press, 2010, pp. 391–397.