Recognizing Characters with Severe Perspective Distortion Using Hash Tables and Perspective Invariants

Pan Pan, Yuanping Zhu, Jun Sun and Satoshi Naoi
Fujitsu R&D Center Co., Ltd., Beijing, China
{*ppan, yuanping.zhu, sunjun, naoi*}*@cn.fujitsu.com*

***Abstract*—In this paper, we present a novel method to recognize characters with severe perspective distortion using hash tables and perspective invariants. The proposed algorithm consists of storage and voting stages. With the help of perspective invariants, the combinations of 4-tuple bases for the perspective invariant coordinate system are searched out in an efficiently way. The bases are further selected so that the resulting transformation is effective. The characters' features under the perspective invariant coordinate system determine an entry in a one dimensional hash table, which is applied for storage and retrieval. Experimental results show the superior performance of the proposed method in comparison to other existing methods.**

***Keywords*-character recognition; severe perspective distortion; hash table; perspective invariant;**

I. Introduction

Character recognition under severe perspective distortion is an important topic because recognizing such distorted characters is the foundation of many applications, for example real-scene character recognition. In order to resolve the problem of perspective distortion, one category of solution is to first rectify the distorted image to a frontoparallel view and apply traditional recognition method on the rectified image. However, these methods rely on the constraints which are based on particular applications, for example the existing of boundary and text lines [1] [2], quadrilateral edges of the text area [3], or some certain structure [4]. Therefore, in this paper, we focus on a more fundamental category of solution, which is to recognize each character directly.

In [5], structural invariants such as ascender and descender are employed to classify English characters into a reduced symbol set. However, the performance of detection of such invariants highly depends on the contextual information, e.g. the length of text lines. In [6], a descriptor named cross ratio spectrum is proposed and the recognition process involves the comparison of the character's spectrum along with all the templates' spectrum. Although the results are promising, the processing time of this method grows linearly with the number of classes, and thus it limits the application of such method in large data sets.

Geometric Hashing (GH) [7] [8] is a general technique for model-based object recognition even when the objects have undergone an arbitrary transformation or when only partial information is present. The strength of this technique is in its efficiency, in its capability for a straightforward parallel implementation, and in its ability to operate in the presence of partial information. GH technique has been used in affine invariant object recognition [9] and 3D object recognition [10]. In the area of character recognition, Iwamura et al. [11] improve GH and propose a real-time recognition algorithm for camera-captured characters. Affine transformation model is used, where a 3-tuple basis is required to construct an affine invariant coordinate system. Affine invariants, i.e. center of gravity and area ratio, are used to reduce the degrees of freedom of the 3-tuple basis. Although the work [11] shows very promising results, the affine transformation model used could only be considered as an approximation of the perspective transformation when the object size is small as compared with the distance between the object and the camera, i.e. the perspective distortion is small. When the perspective distortion is severe, the approximation is no longer valid. Moreover, center of gravity and area ratio are not invariants under any perspective transformation. Therefore, new algorithm needs to be proposed for recognizing characters with severe or any perspective distortion.

In this paper, we present a novel method to recognize characters with severe perspective distortion using hash tables and perspective invariants. The algorithm which consists of storage and voting stages is proposed in Section II. With the help of perspective invariants, the combinations of 4-tuple bases for the perspective invariant coordinate system are searched out in an efficient way. The bases are further selected so that the resulting transformation is effective. The characters' features under the perspective invariant coordinate system determine an entry in a one dimensional hash table, which is applied for storage and retrieval. Experimental results show the superior performance of the proposed method compared with other existing methods in Section III.

II. Character Recognition under Severe Perspective Distortion

In order to recognize characters with severe perspective distortion, we propose a novel algorithm, with its summary shown in Table I. We assume that each character could be segmented and binarized nicely, and therefore binarization and segmentation are not the topics to be explored in this

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.116

Table I
THE SUMMARY OF THE PROPOSED ALGORITHM

Learning of model characters (storage stage)
For each model character:
1. Binarization.
2. Extract a set of ordered 4-tuple bases.
3. For *each* ordered 4-tuple basis
   (a) Obtain the homography matrix which transforms the model character to the perspective invariant coordinate system, and check if the basis is valid.
   If the basis is valid
   (b) Transform the model character to the perspective invariant coordinate system determined by the 4-tuple basis.
   (c) Insert a hash table entry (Table IV).
   Otherwise do nothing

Recognition characters with perspective distortion (voting stage)
For each character to be recognized:
1. Load in the hash table.
2. Binarization.
3. Extract a set of ordered 4-tuple bases.
4. For *some of* the ordered 4-tuple basis
   (a) Obtain the homography matrix which transforms the character to the perspective invariant coordinate system, and check if the basis is valid.
   If the basis is valid
   (b) Transform the character to the perspective invariant coordinate system determined by the 4-tuple basis.
   (c) Perform voting (Table IV).
   Otherwise do nothing
5. The class which receives the highest vote is the recognition result.

paper. Please note that our algorithm is not a straightforward application of general Geometric Hashing technique [7] for perspective transformation, nor it is a trivial extension of camera-captured character recognition with affine transformation model [11]. The differences will be explained in greater detail in the following related sections.

As shown in Table I, the proposed algorithm consists of two stages. In the first stage, the learning (storage) stage, which is performed offline and is independent of the recognition data, the character templates are used to generate a hash table. The hash table contains a description of all the characters that we want to recognize. In the second stage, the recognition (voting) stage, this hash table is used to efficiently vote for the candidate matches.

A. Ordered 4-tuple Bases

In the proposed algorithm, a perspective invariant coordinate system is needed so that the characters under different degrees of perspective distortion could be compared under the same framework. A perspective invariant coordinate system is determined uniquely by a ordered 4-tuple basis (P_0, P_1, P_2, P_3), where $P_i = (x_i, y_i)$ are the horizontal, vertical coordinates of the i^{th} basis point respectively. The problem lies in how to select the bases from characters and how to select them efficiently.

The conventional geometric hashing algorithm is difficult to be used for characters with severe perspective distortion, due to the following two reasons. Firstly the characters are lack of full perspective invariant features. Secondly the computational complexity is high if going through all possible

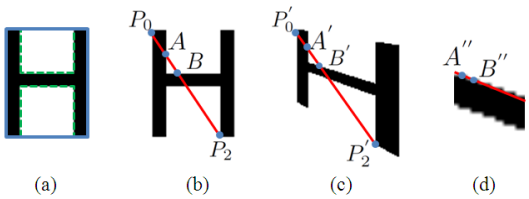


Figure 1. To determine a point using cross ratio. (a) the blue solid line denotes the *convex hull polygon*, and the green dotted line show the *interior contour*. (b)-(c) are examples to show given P_0 and P_0', P_2 and P_2' could be determined. (d) shows the jagged interior contour.

4-tuple combinations one by one. In the previous related work, the points to form the bases are usually feature points, such as corner points [10], edge-triple feature features [12], and external contours of a character's CC [11]. However, those features are not available for separated characters, nor robust for characters with severe perspective distortion. Let use *convex hull polygon* to denote the polygon which vertex are the convex hull of the point set. Since the convex hull of a point-set is preserved under perspective transformation [13] and two distinct points determine a unique line for projective plane [14], we choose the 4-tuple points from the pixels on the boundary of convex hull polygon. As shown in Fig. 1(a), the blue solid line describes the *convex hull polygon*.

The most straightforward way to find out the 4-tuple bases is to go through all the 4-tuple combinations out of all the pixels on the boundary of the convex hull polygon. However, it brings huge burden on computation and storage. For a convex hull polygon with 220 boundary pixels, there are $4C_{220}^4 (\approx 3.8 \times 10^8)$ possible combinations of 4-tuple bases. Therefore, the reduction of degrees of the freedom of the 4-tuple basis is very necessary for the real application on PC. In [11], center of gravity and area ratio are used to reduce two degrees of freedom for a 3-tuple basis of an affine-invariant coordinate system. However, center of gravity, area ratio, SIFT features, maximum curvature point [15] are not perspective invariants. The perspective invariant zeros of curvature of curves [16] could not be used in our problem since in general very few of them. And p^2-invariants consisting of 5-tuple points [13] also could not be used in our framework.

We then use the perspective invariant cross ratio [14] to reduce the computational complexity. Cross ratio has many formats, such as four collinear points, five points, two line two point. And we focus on the format of four collinear points here. Given ordered points (P_0, A, B, P_2) are collinear, the cross ratio $Cr(P_0, A, B, P_2)$ is defined as:

$$Cr(P_0, A, B, P_2) = \frac{D(P_0, B)D(A, P_2)}{D(A, B)D(P_0, P_2)}, \qquad (1)$$

where $D(A, B)$ denotes the distance between points A and B.

We then show that given points P_0 and P_2 from the boundary of the convex hull polygon, P_2 could be determined uniquely by P_0 regardless of perspective transformation, by the help of cross ratio.

Table II
ORDERED 4-TUPLE BASES

P_0, P_1, P_2, P_3 are the points on the boundary of the convex hull polygon.
Step 1: P_0 is an arbitrary point.
Step 2: P_2 is determined by P_0 such that $min\|1.5 - Cr\|$.
Step 3: P_1 is an arbitrary point, such that P_0, P_1, P_2 are counter-clockwise.
Step 4: P_3 is determined by P_1 such that $min\|1.5 - Cr\|$.

Proof: We first prove that cross ratio could be used to determine a point for frontoparallel characters. As shown in Fig. 1(b), P_0, P_2 are the points on the convex hull polygon. Segment P_0P_2 will have intersections with the interior contour. We denote the first two intersections starting from P_0 as A, B. (P_0, A, B, P_2) will determine a cross ratio value as shown in (1). Given point P_0, point P_2 could also be determined by setting a cross ratio value $Cr(P_0, A, B, P_2)$. In order to search P_2 for all character classes, the most widely used criterion is to find P_2 such that $Cr(P_0, A, B, P_2)$ is maximum. However, P_0P_2 may have false intersections with the jagged interior contour (see Fig. 1 (d)), which leads a outlier peak of the cross ratio value. Therefore, the maximum $Cr(P_0, A, B, P_2)$ is not a good choice. Noticing most of the $Cr(P_0, A, B, P_2)$ lies within the range $(1, 2]$, we choose the criterion that $|1.5 - Cr(P_0, A, B, P_2)|$ is minimum.

We then prove that the above statement is true for characters which undergo perspective transformation. Fig. 1 (c) is the perspective distorted Fig. 1 (b). In Fig. 1 (b), P_2 is determined by P_0 and $min|1.5 - Cr(P_0, A, B, P_2)|$ criterion. And in Fig. 1 (c), $P_2^{'}$ is determined by $P_0^{'}$ and $min|1.5 - Cr(P_0^{'}, A^{'}, B^{'}, P_2^{'})|$ criterion. A, B $(A^{'}, B^{'})$ are the first two intersections of segment P_0P_2 $(P_0^{'}P_2^{'})$ and its interior contour. Since two distinct lines determine a unique point for projective plane [14], a line of P_0P_2 $(P_0^{'}P_2^{'})$ will determine uniquely intersections A, B $(A^{'}, B^{'})$ for the same character. If the interior contours are curves instead of segments, since in digital images, curves are actually made of line segments, the above statement is also true. Therefore, if $P_0^{'}$ corresponds to P_0, $P_2^{'}$ corresponds to P_2. ■

Please note that our usage of cross ratio is different from that in [6]. [6] employs cross ratio to form a spectrum as a descriptor for a point, while in our work cross ratio is employed to locate one point given another.

Therefore, we propose an algorithm to find the ordered 4-tuple bases with two degrees of freedom, shown in Table II. In implementation, the counter-clockwise condition could be checked efficiently by signed area [17]. The proposed way to search out the 4-tuple bases not only lower down the computational cost at the stage of both learning and recognition, but also reduce the size of the hash table.

B. Perspective Invariant Coordinate System and Valid Bases

In Table I, a perspective invariant coordinate system is needed so that the characters under different degrees of perspective distortion could be compared fairly. We propose to construct a perspective invariant coordinate system shown in Table III. The coordinate frame is constructed as a square with side length L and center (x_c, y_c). P_0, P_1, P_2, P_3 in the images are matched to I_0, I_1, I_2, I_3 respectively in the perspective invariant coordinate system, where $I_0 = (x_c - l, y_c - l)$, $I_1 = (x_c - l, y_c + l)$, $I_2 = (x_c + l, y_c + l)$, $I_3 = (x_c + l, y_c - l)$, and $l = \alpha L/2$, $\alpha \in [0, 1]$. The matched four pairs will determine a uniquely homography matrix, which could transform the images to the perspective invariant coordinate system.

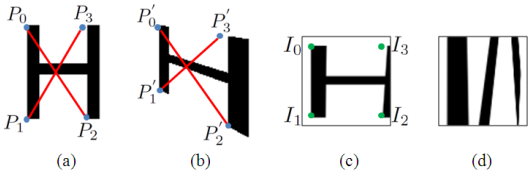


Figure 2. The perspective invariant coordinate system. (c) is the transformed (a) (b) under perspective invariant coordinate system, where P_0, P_1, P_2, P_3 and $P_0^{'}, P_1^{'}, P_2^{'}, P_3^{'}$ are chosen as bases respectively. (d) is one case that transformation is made according to an invalid basis, i.e. less than 90% of the convex hull polygon points are mapped inside of the coordinate frame. Therefore, (d) is ignored during both learning and recognition.

Table III
PERSPECTIVE INVARIANT COORDINATE SYSTEM AND VALID BASES

The coordinate frame is a square with side length L and center (x_c, y_c).
• A 4-tuple basis P_0, P_1, P_2, P_3 corresponds to $(x_c - l, y_c - l)$, $(x_c - l, y_c + l)$, $(x_c + l, y_c + l)$, $(x_c + l, y_c - l)$, respectively, where $l = \alpha * L/2$, $\alpha \in [0, 1]$. The four matching pairs will determine a unique perspective transformation matrix H.
• If $\beta \in [0, 1]$ or more of the points on the convex hull polygon are transformed within the $L \times L$ square, this basis is considered as *valid*.

For an effective storage and voting, we hope significant portion of characters lie within $L \times L$ perspective invariant frame after transformation. Therefore we need to check whether one 4-tuple basis is *valid* or not. We then propose that if $\beta \in [0, 1]$ or more of the points on convex hull polygon of the images are transformed within the $L \times L$ square, this 4-tuple basis is *valid*. Take Fig. 2 as an example, P_0, P_1, P_2, P_3 in Fig. 2 (a) correspond to $P_0^{'}, P_1^{'}, P_2^{'}, P_3^{'}$ in Fig. 2 (b). And they are both transformed to I_0, I_1, I_2, I_3 in Fig. 2 (c). Therefore given the corresponding bases, the transformed images of Fig. 2 (a) and Fig. 2 (b) are the same, shown in Fig. 2 (c). Fig. 2 (d) is an example of the transformation with an invalid basis. In implementation, we choose L=41 so that the transformation is adequately fast, and $\alpha = 0.8$, $\beta = 0.9$.

C. Storage and Voting

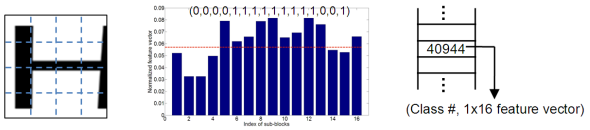
The detailed algorithm for storage and voting is listed in Table IV. At the storage stage, after the template are transformed into the perspective invariant coordinate system, we extract features, and the feature vector is used to determine a hash table bin. We rely on a simple feature, the block histogram of black pixels [11]. After all templates are

Table IV
STORAGE AND VOTING

The perspective invariant coordinate frame is divided into $m \times m$ sub-squares.
- Generate a one dimensional histogram where the x-axis is the number of sub-squares, and the y-axis is the number of black pixels.
- Feature vector $\mathbf{f_{current}}$ is the normalized histogram.
- Feature vector is quantized and the quantized vector is mapped to a decimal bin number b (from n-ary to decimal).

Storage
- Insert an entry in the hash table (bin, class, feature vector).

Voting
- Retrieve the class number d and feature vector $\mathbf{f_{stored}}$ from bin b.
- If $\|\mathbf{f_{current}} - \mathbf{f_{stored}}\| < C$
  Give a vote to class d; ignore the duplicate votes from bin b to the same class d.
- Otherwise do nothing.



(a) (b) (c)

Figure 3. Storage and voting. (a) the perspective invariant coordinate system is divided into 4×4 sub-squares. (b) The feature vector is a normalized histogram of black pixels in different blocks. The feature vector is further quantized into 2 levels. (c) The quantized feature vector is mapped to a decimal number, which is the bin number in the hash table.

processed one by one, we obtain a one dimensional hash table, which has the following structure: (bin, class, feature vector). Figure 3 shows an example. Bases tuples could also be stored, and may be used in a verification step during recognition [8].

At the voting stage, the first three steps are the same. After the bin is located, we give one vote to the class number in that bin if the feature is close to the stored feature. And the duplicate votes from bin b to the same class d are ignored. In the end, the class which receives the largest number of votes is the recognition result. Since the final result is based on voting, there is no need to go through all the possible combinations of 4-tuple bases, but sufficient combinations are enough.

III. EXPERIMENTAL RESULTS

In order to demonstrate the performance of our proposed algorithm, we conduct experiments on test data. We employ 62 characters, namely 26 uppercase English characters, 26 lowercase English characters and 10 digits, of Arial font. Screen print frontoparallel characters are utilized as the templates for learning. Since some of the characters are difficult to be distinguished under perspective transformation, the characters in a cell in Table V are treated as one class in the experiments. We compare the proposed algorithm with the algorithm using affine transformation model [11]. For both methods, at recognition stage the bases points which have degrees of freedom are sampled from every other points on the convex hull polygon (external contour for affine model

Table V
LIST OF SIMILAR CHARACTERS. CHARACTERS IN A CELL ARE TREATED AS THE SAME CLASS.

Cc	Iil	Kk	LVv	NZz	0Oo	Ss
Ww	Xx	bq	dp	un	69	

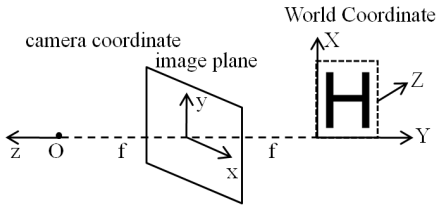


Figure 4. The projective geometry of generating test data.

algorithm [11]), and $m = 4$, $C = 0.11$.

The test data are generated using the perspective model shown in Fig. 4. O is the camera origin and we set the origin of the world coordinate at $(0, 0, -2f)$ of the camera coordinate. Without loss of generality, the frontoparallel view character plane correspond to the X, Y plane of the world coordinate system. The rotation matrix from the world coordinate to the camera coordinate will determine the degree of perspective distortion. In practice, after the image is projected onto the image plane, the character region is extracted and rescaled properly to 100×100. For each character, we increase the tilt and pan of rotation matrix separately, so that we obtain a series of perspective distorted images for which the distortion is from small to severe. Since the origin of world coordinate is close to that of the camera coordinate, a small degree of rotation will cause relative large perspective distortion. When mapping the test data, we do not use any interpolation, and thus we obtain noisy test data. We therefore obtain 1240 test images, for each character with 10 images of tilt changes, and 10 images of pan changes. Figure 5 provides some samples of test images. The recognition rate of both comparative methods are shown in Table VI. The proposed algorithm have higher recognition rate than the method which uses affine model. Figure 6 shows the error rate along with the degree of perspective distortion for comparative methods. We could see that when the perspective distortion is small, both methods perform well. However, the affine model method perform worse than ours when the perspective distortion is large. The experimental observations comply with the fact that the affine model could only be treated as an approximation of the perspective model when the perspective distortion is small.

We also apply the proposed algorithm on real camera-captured text images with severe perspective distortion. Fig. 7 provides some sample characters that our proposed algorithm could successfully recognize while the affine model method [11] fails. Since the recognition step of our method is based on voting, it could still handle some cases when only partial information is present. As shown in Fig. 8, our proposed algorithm could still recognize those characters with missing parts highlighted by dotted red ellipses.

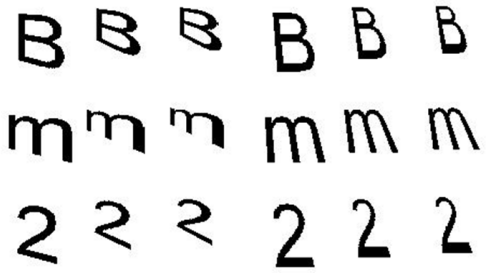


Figure 5. Samples of test data.

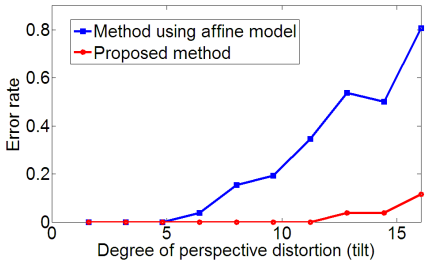


Figure 6. Error rate with respect to the degree of perspective distortion (tilt) of our proposed method and the method using affine model [11].

Figure 7. Samples of camera-captured characters. The proposed method could recognize the above characters correctly, while the affine model method [11] fails.

Figure 8. Samples of characters that our proposed algorithm could recognize correctly, where partial information is present.

For the generated test data, the recognition time for a character is 3.78 seconds on average. The time is measured on an Intel Core Duo 2.4G computer without code optimization. Since the time consuming steps of our proposed algorithm are in parallel format, the processing time could be greatly reduced using parallel programming techniques, which is one of our future work. Please also note that other techniques, e.g. the generative learning of templates [11], could be incorporated into our framework, which will further improve the recognition results on degraded characters.

IV. Conclusion

In this paper, we proposed a novel method to recognize characters with severe perspective distortion using hash tables and perspective invariants. Cross ratio is used to lower down the complexity to search out the combinations of 4-tuple bases. The bases are further selected so that the resulting transformation under the perspective invariant coordinate system is effective. A one dimensional hash table which corresponds to the features under the perspective invariant coordinate frame is applied for storage and voting. Experiments show the better performance of the proposed method compared with other methods. Our proposed algorithm could not only achieve higher recognition accuracy on characters with severe distortions, but also have the ability to operate when partial information is present and the capability for a straightforward parallel implementation.

Table VI
Recognition rate of comparative methods

Proposed method	94.11%
Method using affine model	70.00%

References

[1] M. Pilu, "Extraction of illusory linear clues in perspectively skewed documents," in *CVPR*, 2001.

[2] X.-C. Yin, J. Sun, and S. Naoi, "A multi-stage strategy to perspective rectification for mobile phone camera-based document images," in *ICDAR*, 2007.

[3] P. Clark and M. Mirmehdi, "Recognizing text in real scenes," *Int. J. Document Analysis and Recognition*, vol. 4, no. 4, pp. 243-257, 2004.

[4] M. Lourakis, "Plane metric rectification from a single view of multiple coplanar circles," in *ICIP*, 2009.

[5] S. Lu and C. L. Tan, "Camera text recognition based on perspective invariants," in *ICPR*, 2006.

[6] L. Li and C. L. Tan, "Character recognition under severe perspective distortion," in *ICPR*, 2008.

[7] Y. Lamdan and H. J. Wolfson, "Geometric hashing: a general and efficient model-based recognition scheme," in *ICCV*, 1988.

[8] H. J. Wolfson and I. Rigoutsos, "Geometric hashing: an overview," *IEEE Computational Science and Engineering*, vol. 4, no. 4, pp. 10-21, 1997.

[9] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson, "Affine invariant model-based object recognition," *IEEE Trans. Robotics and Automation*, vol. 6, no. 5, pp. 578-589, 1990.

[10] H. v. Dijck and F. v. d. Heijden, "Object recognition with stereo vision and geometric hashing," *Pattern Recognition Letters*, vol. 24, pp. 137-146, 2003.

[11] M. Iwamura, T. Tsuji, A. Horimatsu, and K. Kise, "Real-time camera-based recognition of characters and pictograms," in *ICDAR*, 2009.

[12] S. Procter and J. Illingworth, "Foresight: fast object recognition using geometric hashing with edge-triple features," in *ICIP*, 1997.

[13] P. Meer, R. Lenz, and S. Ramakrishna, "Efficient invariant representations," *Int. J. Computer Vision*, vol. 26, no. 2, pp. 137-152, 1998.

[14] J. Mundy and A. Zisserman, *Geometric Invariance for Computer Vision*. MIT Press, 1992, ch. Appendix – Projective Geometry for Machine Vision.

[15] P. Putjarupong, C. Pintavirooj, W. Withayachumnankul, and M. Sangworasil, "Image registration exploiting five-point coplanar perspective invariant and maximum-curvature point," in *Int. Conf. in Central Europe on Computer Graphics, Visualization and Computer Vision*, 2004.

[16] A. Verri and A. Yuille, "Perspective projection invariants," Massachusetts Institute of Technology, Tech. Rep., 1986.

[17] W. H. Beyer, *CRC Standard Mathematical Tables*. CRC Press, 1987.