Pseudo-Zernike Moments for Feature Extraction and Chinese Character Recognition

Peter Kasza [mr.schyte@gmail.com]

***Abstract*—In this paper I present a pseudo-Zernike feature descriptor based recognition technique for accurate identification of printed and handwritten Chinese characters. Chinese character recognition is a hard topic amongst other character recognition problems not only because the sheer number of different characters to identify, but also because of the characters' dense and complex nature. Moment functions have been successfully applied to many shape recognition problems, due to the fact that they tend to capture global features, which makes them well suited as feature descriptors. However simple moment based techniques fail at capturing the different features of structurally complex objects, such as Chinese characters. To solve this problem more efficient moment functions have to be used.**

***Index Terms*—Image processing, Geometric moments, Zernike moments, pseudo-Zernike moments, Orthogonal Moments, Feature descriptors, Invariance features of moments, Character recognition, Shape recognition, Machine learning, Neural networks, Chinese characters**

I. Introduction

Moment based techniques have been successfully applied to several image processing problems and they represent a fundamental tool for generating feature descriptors. Feature descriptors built from moment functions capture global features, and as such they are well suited for shape and character recognition. Some moment functions exhibit natural invariance properties, such as invariance to rotation, translation or scaling. Translation and scale invariance is usually obtained by normalizing the input image with its geometric moments. The original image can be reconstructed from its moments, as an infinite series of moments uniquely identify a specific distribution. Reconstruction from orthogonal moment functions have been discussed in detail, while recovery methods for non-orthogonal moments, such as geometric moments have only been proposed recently in works such as [1]. By reconstructing the original image intensity function, not only the implementation's correctness can be tested, but also the error-rate of the calculation and the optimal number of moments can be identified.

Because Zernike moments are defined over the unit disk, they are naturally uneffected by rotation. The calculation of Zernike moments is a computationally intensive problem, substantiated by the fact, that they have a very large dynamic range. Zernike moments share little information between each other and as such, they can be used to efficiently represent global features of images. Pseudo-Zernike moments can be thought of as an augmented version of Zernike moments. Pseudo-Zernike moments contain around twice as much moments of order $(p+q)$ than Zernike moments, with both sharing the orthogonality property, allowing efficient recovery of the original image. Pseudo-Zernike moments have been shown to be superior to Zernike moments in terms of sensitivity to noise and their overall error-rate. [1]

Character recognition in general, can be categorized as handwritten and printed character recognition. Handwritten character recognition is considered to be a harder topic, as the characters' shape contain far more irregularities than their printed counterparts. Character and shape recognition builds upon global features rather than local ones. Global features tend to represent the actual shape of the object, as opposed to local features, which usually capture a combination of non-useful information, such as stroke width and noise.

Chinese character recognition is an especially hard topic in character recognition. Chinese characters have a very dense structure containing many strokes, thus using simple feature descriptors for identification is ineffective. Some characters only differ in small detail, yet posess different meanings. To further complicate the problem, there is a very large number of Chinese characters, obtaining an input dataset of sufficient size is a challenging task. Printed characters are usually written in regular script utilizing mostly diagonal lines, while handwritten versions tend to appear in cursive, or semi-cursive form.

II. Moment functions

Moment functions are defined on images as the weighted sums of the image intensity function. Moment functions of order $(p+q)$ are generally defined as

$$\phi_{pq} = \int_x \int_y \psi_{pq}(x,y) f(x,y)\, dx dy,$$

where $\psi_{pq}(x,y)$ is called the moment weighting kernel.

When applying moment functions to digital images it is often desirable to write them out using the following discrete notation:

$$\phi_{pq} = \sum_x \sum_y \psi_{pq}(x,y) f(x,y).$$

Some properties of the weighting kernel are passed onto the moments themselves, such as invariance features, and orthogonality. Depending on the function chosen for the weighting kernel, the calculated moments can capture different aspects of the input image.

A. Geometric moments

Geometric moments are defined as follows:

$$m_{pq} = \int_x \int_y x^p y^q f(x,y)\, dx dy.$$

Notice that the weighting kernel $\psi_{pq}(x,y) = x^p y^q$ is not orthogonal, and as such, the original image function cannot be easily obtained from a generated set of geometric moments. However Fourier transformation based techniques exist for accomplishing this task as described in [1]. Using geometric moments, several important features of the image function can be described, most importantly the centre of gravity for the image.

B. Zernike moments

As opposed to geometric moments, Zernike moments are defined over the unit disk instead of the real plane and exhibit the orthogonality property. Zernike polynomials are mainly used in optometrics, where they arise as the expension of a wavefront function in optical systems with circular pupils. [5]

Zernike moments are defined using Zernike polynomials as follows:

$$Z_{pq} = \frac{(p+1)}{\pi} \int_0^{2\pi} \int_0^1 V^*_{pq}(r,\varphi) f(r,\varphi) r \, dr d\varphi,$$

where:

$$r \leq 1, \quad |q| \leq p, \quad p - |q| \text{ is even.}$$

The Zernike polynomial $V_{nm}(r,\varphi)$ is defined as a function of the radial polynomial:

$$V_{nm}(r,\varphi) = R_{nm}(r) e^{im\varphi},$$

and the radial polynomial is:

$$R_{nm}(r) = \sum_{s=0}^{(n-|m|)/2} \frac{r^{n-2s}(-1)^s (n-s)!}{s! \left(\frac{n-2s+|m|}{2}\right)! \left(\frac{n-2s-|m|}{2}\right)!}.$$

It is often more convenient to write the radial polynomial in the following coefficient form:

$$R_{nm} = \sum_{k=m}^{n} B_{nmk} r^k, \quad (m \geq 0,\, n-k \text{ is even})$$

with B_{nmk} defined as:

$$B_{nmk} = \frac{(-1)^{(n-k)/2} \left(\frac{n+k}{2}\right)!}{\left(\frac{n-k}{2}\right)! \left(\frac{k+m}{2}\right)! \left(\frac{k-m}{2}\right)!}.$$

C. pseudo-Zernike moments

Pseudo-Zernike moments have been shown to be superior to Zernike and Hu moments as feature descriptors in terms of their exhibited error rate. [4] However due to their costly computation and high dynamic range, pseudo-Zernike moments haven't seen wide use in image recognition yet. Pseudo-Zernike moments of order $(p+q)$ contain twice as much information as Zernike moments, because the restriction that $p - |q|$ is even, does not exist.

Pseudo-Zernike moments can be tought of as an augmented version of Zernike moments. They inherit the same orhtogonality property, but instead of the radial polynomials pseudo-Zernike moments use the following:

$$S_{nm}(r) = \sum_{s=0}^{n-|m|} \frac{r^{n-s}(-1)^s (2n+1-s)!}{s!(n-|m|-s)!(n+|m|+1-s)!}$$

We can write them in the same form as Zernike moments using the pseudo-Zernike polynomial $\tilde{V}_{pq}(r,\varphi)$:

$$\tilde{Z}_{pq} = \frac{(p+1)}{\pi} \int_0^{2\pi} \int_0^1 \tilde{V}^*_{pq} f(r,\varphi) r \, dr d\varphi, \quad r \geq 1,$$

where:

$$\tilde{V}_{pq} = S_{nm}(r) e^{iq\varphi}.$$

We can write S_{nm} in terms of its coefficients the same way as with radial polynomials. Written in coefficient form:

$$S_{nm}(r) = \sum_{k=m}^{n} C_{nmk} r^k$$

with:

$$C_{nmk} = \frac{(-1)^{n-k}(n+k+1)!}{(n-k)!(k+m+1)!(k-m)!}.$$

Using the following recurrence relations the coefficients can be calculated more efficiently [2]

$$C_{nnn} = 1 \tag{1}$$

$$C_{n(m-1)k} = C_{nmk} \frac{k+m+1}{k-m+1} \tag{2}$$

$$C_{nm(k-1)} = -C_{nmk} \frac{(k+m+1)(k-m)}{(n+k+1)(n-k+1)} \tag{3}$$

I have found that calculating k-th power of r takes the longest time, compared to the coefficients, which can also be cached using a dynamic programming approach. Due to the high dynamic range of pseudo-Zernike and Zernike moments, arbitrary precision libraries must be used for calculating up to even moderatly small orders. As pseudo-Zernike moments are defined over the unit disk, cartesian images must be converted to polar representation, and normalized first.

Converting to polar coordinates can be achieved by applying the following coordinate transformations:

$$\varphi = \tan^{-1}\left(\frac{y}{x}\right), \quad r = \|x+y\|_2 / r_{max}.$$

D. Invariance properties

Translation invariance of pseudo-Zernike moments can be achieved by translating the input image by its geometric moments, so that both m_{01} and m_{10} is zero on the resulting image. Similarily scale invariance is achieved by normalizing the image, so that the total area of foreground pixels is of a predetermined value β. Because pseudo-Zernike moments are defined on the unit disk, they are unaffected by rotation, as rotating the image only changes the phase angle of moments.

Rotation by an angle of ϕ can be written as:

$$\tilde{Z}^R_{pq} = \tilde{Z}_{pq} e^{-iq\phi}.$$

Using the following transformation, both translation and scale invariance can be achieved, where β is an arbitrary constant value, specifying the overall area of the normalized image. [3]

$$g(x,y) = f\left(\frac{x}{a} + \overline{x}, \frac{y}{a} + \overline{y}\right),$$

where:

$$\overline{x} = \frac{m_{10}}{m_{00}},\ \overline{y} = \frac{m_{01}}{m_{00}},\ a = \sqrt{\frac{\beta}{m_{00}}}$$

E. Inverse moment problem

Using the Fourier inversion theorem for othogonal bases, the original image function can be written as:

$$f(x,y) \cong \sum_{p=0}^{n} \sum_{q} \tilde{Z}_{pq} \tilde{V}_{pq}(r,\varphi), \quad |q| \leq p.$$

Because the image function's values are from the real number plane, we can safely assert that the imaginary part of the function value is always zero for any input. Due to rounding errors however, if we would calculate the imaginary part we probably won't get a value of zero. The absolute sum of the recovered function's imaginary part can be a good indicator of the rounding errors' cumulative magnitude.

$$\operatorname{Im} f(x,y) = 0$$

Using Euler's law, we can write $\tilde{Z}_{pq}$ in the following form:

$$\tilde{Z}_{pq} = |\tilde{Z}_{pq}| e^{i \arg(\tilde{Z}_{pq})}$$

Then using the law of exponents and taking the real part gives:

$$\operatorname{Re} f(r,\varphi) \cong \sum_{p=0}^{n} \sum_{q} |\tilde{Z}_{pq}| S_{pq}(r) \cos(\varphi q \arg(\tilde{Z}_{pq})).$$

Which is the main equation for the inverse transformation.

III. Feature extraction

Due to the fairly large number of Chinese characters, I have decided to only use the standard 1945 Japanese joujou kanji, which most freely available TrueType fonts support. The input dataset I used contains 40592 images in total, with about 20 differently rendered variants for each character. The images are of 32×32 pixels in size, with pseudo-Zernike moments calculated up to the 32th degree.

A. Training the Neural Network

The hardest problem in designing neural networks, is figuring out their layer structure. Even though algorithms exists for the automatic configuration of the layer structure during training, in our case the number of input samples makes such approaches impossible, as training would take a very long time. Instead the configuration had to be found using parameter tweaking, and partially training the network. As a result a standard 3-layer backpropagation network was used, having 16×16 input values, 256 neurons in the hidden layer and 1945 outputs.

Another important design consideration for neural networks is how to represent the output values. In our case an obvious solution is to use a unique index identifying each of the input images. However using a single value as the output value has several problems. First of all the network cannot generate values outside the $[0,1]$ or $[-1,1]$ range. Secondly, even if we map the indices into the said region, at most only two possible indices can be generated; those that are closest to the output value. To overcome this problem, we have to use a binary vector representing the appropriate indices. Each element of the output vector specifies whether the current index is the same as the input image. The output vectors can be written in terms of the Kronecker delta function as follows:

$$O_{ij} = \delta_{2^i j}, \quad (j = 1, \ldots, n)$$

where n is the total number of indices and i specifies the current index.

B. Test results

I have found that calculating pseudo-Zernike moments up to the 32th order, gave enough information only for the recognition of simpler *handwritten* kanji, containing less than 10–12 strokes, while *printed* characters were identified almost with a 100% accuracy.

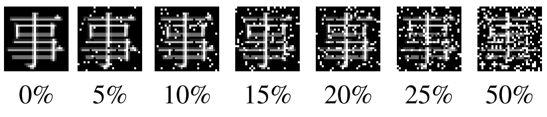

Fig. 1. Chinese characters with different amount of salt and pepper noise

The trained neural network was tested with different levels of salt and pepper noise, using a dataset generated from a TrueType font that the network haven't seen before. Testing shows that the feature descriptors react well, even at a high amount of noise. Figure III-B shows that even at a 50% noise level; at which characters are hard to discriminate even for human observers, the network was able to identify the characters with relative ease.

IV. Conclusion

I have shown that using pseudo-Zernike moments for feature descriptors is a viable option for classifying structurally complex images. They provide excellent invariance features and exhibit better performance than other moment based solutions. Even when the input is stained with heavy amounts of noise, they still provide an accurate method for character identification. The only drawback is their costly computation, which makes them unsuitable for some problems. However they can be calculated in parallel, and as the computational performance of computers increase, the time required for their calculation probably won't be a problem in the near future.

References

[1] Faouzi Ghorbel Stephane Derrode Sami Dhahbi Rim Mezhoud. Reconstructing with geometric moments. In *ACIDCA*. Groupe de Recherche Images et Formes de Tunisie (GRIFT), Laboratoire CRISTAL, Ecole Nationale de Sciences de L'Informatique (ENSI), Campus Universitaire, 2080 La Manouba, Tunisia, 2005.

[2] Mukundan R. Chong C.W. Raveendran P. An efficient algorithm for fast computation of pseudo-zernike moments. In *New Zealand: International Conference on Image and Vision Computing - IVCNZ'01*. University of Canterbury, Computer Science and Software Engineering, Nov 2001.

[3] Jamie Shutler. Statistical moments. CVonline: On-Line Compendium of Computer Vision [Online]. R. Fisher (ed). Available: http://homepages.inf.ed.ac.uk/rbf/CVonline/. [2009, 08. 14], August 2002.

[4] C. H. Teh and R. T. Chin. On image analysis by the methods of moments. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 10(4):496–513, 1988.

[5] Eric W. Weisstein. Zernike polynomial. From MathWorld – A Wolfram Web Resource. http://mathworld.wolfram.com/ZernikePolynomial.html.

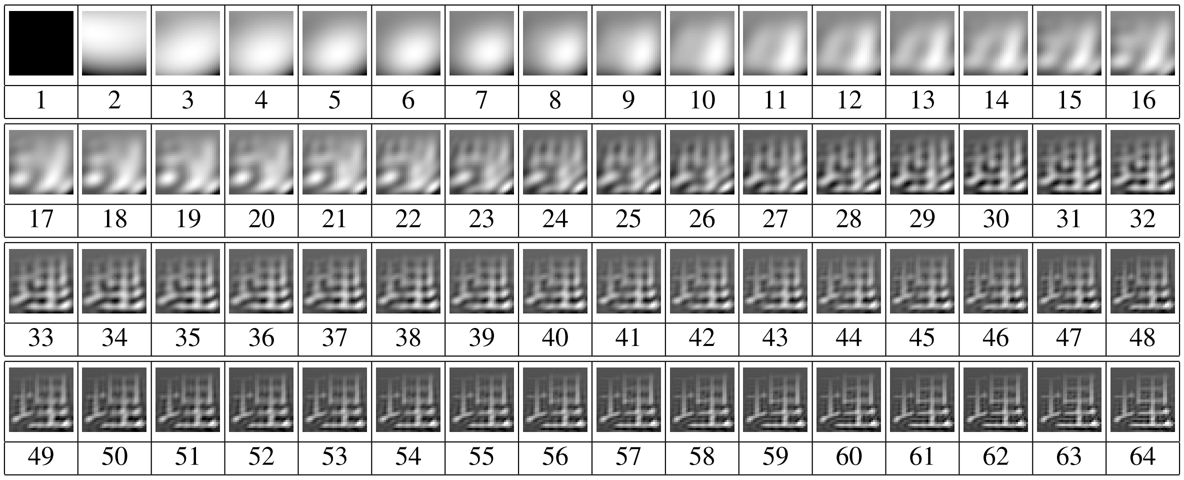

Fig. 2. Recovered images from the nth order pseudo-Zernike moments

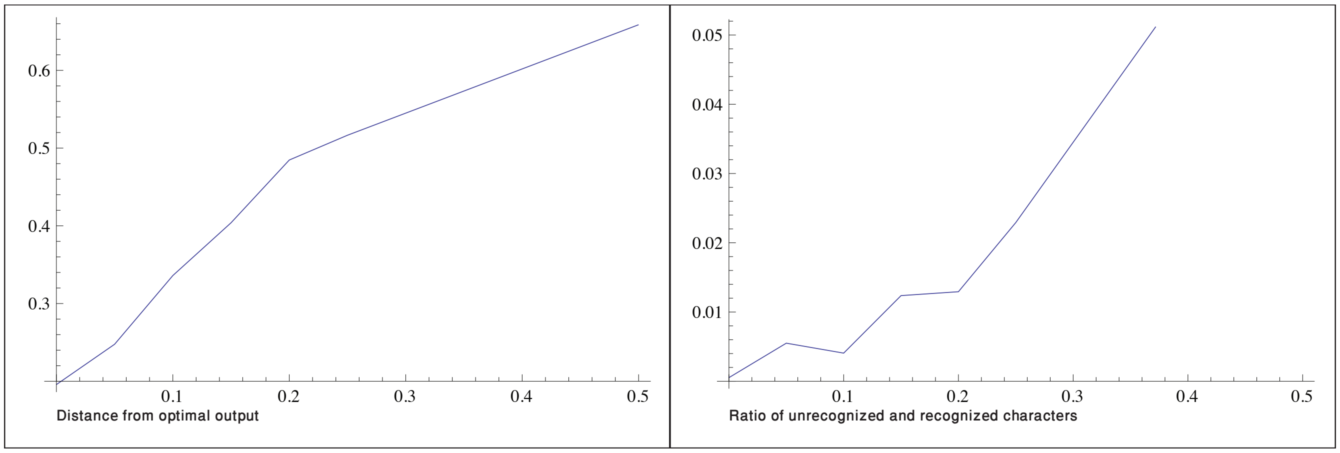

Fig. 3. Statistics genrated from test results

	ratio of unrecognized characters	distance from optimal output	max. distance from best index	mean distance from best index	max. indices per group	mean indices per group
0%	0.000524	0.195760	7	0.130890	15	3.053470
5%	0.005501	0.247628	6	0.412541	15	2.158869
10%	0.004065	0.336000	8	0.463995	15	2.444730
15%	0.012370	0.403924	9	0.886719	14	2.629820
20%	0.012931	0.484647	7	1.199713	14	2.915167
25%	0.022920	0.516534	9	1.511036	16	3.119794
50%	0.080882	0.658699	10	3.966912	17	3.715681

Fig. 4. Statistics genrated from test results