Ratio Mathematica Volume 41, 2021, pp. 53-63

An image compression method based on Ramanujan Sums and measures of central dispersion

Sadanandan Sajikumar *
John Dasan †
Vasudevan Hema ‡

Abstract

This paper introduces a simple lossy image compression method based on Ramanujan Sums $c_q(n)$ and the statistical measures of numerical data such as mean and standard deviation. The Ramanujan Sum $c_q(n)$ has been used in digital signal processing for a variety of applications nowadays. Some of them include the recently developed image kernels for edge detection, extraction of periodicity from signals, etc. The presented compression algorithm is an extension of the edge detection algorithm using an integer image kernel based on Ramanujan Sums. We propose a block-based compression algorithm that detects edges in the images using this image kernel and then compresses the image by storing kernel operation values, the mean and standard deviation for each block instead of pixel values. The proposed method has the advantage of low computational complexity and shows its ability in fast reconstruction and high compression that can be achieved for different block sizes.

Keywords: Ramanujan Sum $c_q(n)$; lossy image compression; mean; standard deviation.

2020 AMS subject classifications: 11Z05; 97M10. [1]

---

*First Author's Affiliation (College of Engineering Trivandrum, Thiruvananthapuram, Kerala, India); sajikumar.s@cet.ac.in.

†Second Author's Affiliation (College of Engineering Trivandrum, Thiruvananthapuram, Kerala, India); dasanj@cet.ac.in.

‡Third Author's Affiliation (College of Engineering Trivandrum, Thiruvananthapuram, Kerala, India); hema@cet.ac.in.

[1]Received on October 28, 2021. Accepted on December 20, 2021. Published on December 31, 2021. doi: doi:10.23755/rm.v41i0.683. ISSN: 1592-7415. eISSN: 2282-8214.

1 Introduction

In the rapid popularization of the internet and social media, the role of a digital image is indispensable. As we have to transmit a lot of information over communication networks, bandwidth reduction is necessary. To achieve this, audio and video signals need to be compressed. Image or video signals in compressed form are convenient for editing, storing, utilizing, and transmitting. Image compression techniques can be classified into two categories. If the information retained after decompression is 100%, it is called lossless compression otherwise lossy. If we take a pixel in an image at random there is a good chance that its neighbours will have the same intensity or very similar intensity. Typically hence, image compression is based on the fact that the neighbouring pixels are highly correlated ([Salomon, 2007], [Sayood, 2012]). Most image compression methods exploit this feature to obtain efficient compression.

Lossless compression can be achieved with the techniques like Run Length Encoding (RLE), Huffman coding, Arithmetic coding etc.([Gallager, 1978], [Jain, 1989], [Taubman and Marcellin, 2012], [Witten et al., 1987]). Lossy techniques include transform coding methods such as Discrete Cosine Transform (DCT), JPEG, JPEG2000 etc.([Pennebaker and Mitchell, 1992], [Gonzalez and Woods, 2008], [Goyal, 2001]). Polynomial-based compression is another lossy compression method ([Sadeh, 1996],[Eden et al., 1986]). Sajikumar S et al., [Sajikumar and Anilkumar, 2017] introduced a compression scheme using Chebyshev polynomials. The proposed compression algorithm differs from the standard compression algorithms in its low computational complexity and fast reconstruction.

Lossy compression techniques are tested for their performance based on three commonly used measures, the Root Mean Square Error (RMSE), Peak Signal to Noise Ratio (PSNR) and the Compression Ratio (CR). The RMSE between original image $f(x,y)$ and reconstructed image $\hat{f}(x,y)$ of size $M \times N$ is defined by [Joshi, 2018]:

$$\text{RMSE} = \sqrt{\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[f(x,y)-\hat{f}(x,y)\right]^2} \tag{1}$$

For an 8- bit gray level image,

$$\text{PSNR } = 10\log_{10}\Big(\frac{255^2}{MSE}\Big)(dB) \tag{2}$$

$$\text{CR} = \frac{\text{compressed image size}}{\text{uncompressed image size}}\ \% \tag{3}$$

An image kernel is a matrix used to obtain effects like blurring, sharpening, outlining, etc. Computer vision applications of image kernel mainly include feature extraction and edge detection. A geometric perspective of kernel methods can be seen in [Lampert, 2009]. Zhang et al., studies various non-local kernel regression for image and video restoration tasks [Zhang et al., 2010]. Odone et al., describes methods for building kernels from binary strings for image matching [Odone et al., 2005]. In 2016, Krishnaprasad P et al., [Krishnaprasad and Ramanujan, 2016] presented an image kernel based on Ramanujan Sums to detect edges. For each 3×3 block of pixels in the image, they multiplied each pixel by the corresponding entry of the 3×3 kernel matrix constructed from $c_3(n)$ and then takes the sum. This sum is considered as a new pixel in the image.

Ramanujan Sums $c_q(n)$ is a family of trigonometric sums defined by Srinivasa Ramanujan in 1918 [Ramanujan, 1918]. In the last fifteen years, Ramanujan Sums have aroused some interest in signal processing. Cohen initially introduced Ramanujan Sums to the signal processing community ([Cohen, 1955], [Cohen, 1958]). In 1950, he observed that the DFT coefficients of even symmetric signals can be computed by integer-valued weighting coefficients. Later it was proved that these integer-valued coefficients are nothing but the well-known Ramanujan Sums.

The rest of the paper is organized as follows. A brief description of the Ramanujan Sum is given in section 2. Image kernel construction and the proposed compression algorithm are given in sections 3 and 4 respectively. Results and discussion are included in section 5 and section 6 concludes the paper.

2 Review of Ramanujan Sums

The Ramanujan Sum $c_q(n)$ has been used by mathematicians to derive many important infinite series expansions for arithmetic functions in number theory [Apostol, 1976]. Interestingly, this sum has many properties which are attractive from the point of view of digital signal processing. Srinivasa Ramanujan defined the q^{th} Ramanujan Sum by

$$c_q(n) = \sum_{\substack{k=1 \\ (k,q)=1}}^{q} W_q^{kn} = \sum_{\substack{k=1 \\ (k,q)=1}}^{q} W_q^{-kn} \tag{4}$$

where $W_q = e^{-i2\pi/q}$, $\quad i = \sqrt{-1}$ and (k, q) denotes the gcd of k and q. Here the sum runs over those k satisfying $(k, q) = 1$ means that we are considering all the integers which are coprime to q in the summation.

For example, if $q = 8$ then $k \in \{1, 3, 5, 7\}$ so that

$$c_8(n) = e^{i2n\pi/8} + e^{i6n\pi/8} + e^{i10n\pi/8} + e^{i14n\pi/8}$$

In number theory, the number of integers less than or equal to q and coprime to q is called the Euler's totient function $\phi(q)$ Apostol [1976]. Since $1, 3, 5, 7$ are coprime to 8, $\phi(8) = 4$.

So the sum given in equation (4) has precisely $\phi(q)$ terms and it is clear that $c_q(0) = \phi(q)$. Also

$$c_q(n+q) = \sum_{\substack{k=1 \\ (k,q)=1}}^{q} e^{i2n\pi k/q}.e^{i2\pi k} = \sum_{\substack{k=1 \\ (k,q)=1}}^{q} e^{i2n\pi k/q} = c_q(n)$$

That is $c_q(n)$ is periodic with period q.

If $(k, q) = 1$, we have $(q - k, q) = 1$. Therefore,

$$(W_q^k)^* = W_q^{-k} = W_q^{-(q-k)} = W_q^k$$

where $*$ is the complex conjugate. This implies that the summation (4) is real valued and it can also be written as :

$$c_q(n+q) = \sum_{\substack{k=1 \\ (k,q)=1}}^{q} \cos \frac{2n\pi k}{q} \tag{5}$$

From (5), $c_q(n) = c_q(-n)$ shows that $c_q(n)$ is symmetric. Thus $c_q(n)$ is a real, symmetric, and periodic sequence in n.

For $0 \leq n \leq q - 1$, first few Ramanujan sequences are

$$\begin{aligned} c_1(n) &= 1 \\ c_2(n) &= 1, -1 \\ c_3(n) &= 2, -1, -1 \\ c_4(n) &= 2, 0, -2, 0 \\ c_5(n) &= 4, -1, -1, -1, -1 \end{aligned}$$

Note that $c_q(n)$ is integer-valued and further properties can be seen in [Vaidyanathan, 2014].

3 Image kernel

Krishnaprasad et al., [Krishnaprasad and Ramanujan, 2016], has introduced a kernel matrix M_q of size $q \times q$ constructed from $c_q(n)$ by considering the circular

shifts of the q elements
$c_q(0), c_q(1), \cdots, c_q(q-1)$ in each row. The first row of the kernel matrix contains the q elements in the order $c_q(0), c_q(1), \cdots, c_q(q-1)$

where $c_q(r) = \sum_{\substack{k=1 \\ (k,q)=1}}^{q} e^{i2\pi kr/q} \qquad \text{for } 0 \leq r \leq q-1.$

The second-row elements are $c_q(q-1), c_q(0), c_q(1), \cdots, c_q(q-2)$ and so on. Thus

$$M_q = \begin{bmatrix} c_q(0) & c_q(1) & c_q(2) & .... & c_q(q-1) \\ c_q(q-1) & c_q(0) & c_q(1) & .... & c_q(q-2) \\ \vdots & \vdots & \vdots & ... & \vdots \\ c_q(1) & c_q(2) & c_q(3) & .... & c_q(0) \end{bmatrix}$$

4 Proposed method

Partition the input image into non-overlapping blocks of size $q \times q$. Test images of size 256×256 with 8- bit gray levels between 0 and 255 are considered. Multiply each pixel in the $q \times q$ block with the corresponding elements of the kernel M_q and take their sum. This sum is stored for edge detection. Represent the entire block of pixel values with this sum obtained. After the edge detection process, we move on to the compression part. In this step, we are computing the mean and standard deviation of each block of pixels to obtain the texture at decompression. Two different ways of compressing an image with the statistical measures of pixel values are proposed.

A. Method 1

For each $q \times q$ block, a block value is computed by adding the kernel multiplication sum which is obtained at the edge detection stage, the mean and standard deviation of each block of pixels. Represent the entire block with this sum at the reconstruction stage. By varying the block size we can compress the image at different compression levels.

B. Method 2

Instead of taking the sum of three quantities to represent each block, consider the kernel sum and the mean value only. Thus we need to store only two values in place of q^2 pixels and hence high compression is achieved.

Experimental results with the test images are given in Tables 1-3 and Figures 1-2. In both methods, no quantization or postprocessing is done at the reconstruction step.

Algorithm

Step 1 Load an input gray image.

Step 2 Partition the image matrix into non-overlapping blocks of size $q \times q$.

Step 3 For each block compute the elementwise product sum with the kernel matrix and the pixel values of the $q \times q$ block. Also find the mean and standard deviation of the q^2 pixels. Store these values for the reconstruction of each block.

Step 4 Replace the q^2 gray values by the elementwise product sum computed in *Step 3* to detect edges.

Step 5 Replace the q^2 gray values by the sum of the three quantities stored in *Step 3* to compress the image.

OR

Replace the q^2 gray values by the sum of the elementwise product sum and the mean to achieve high compression.

5 Results and Discussion

Edge detection results for the test images of different block sizes 2×2, 3×3, 4×4 are given in Figure 1. Edge detection with 2×2 blocks shows better results as compared to others.

In the first method , we replace q^2 pixel values with three quantities. Hence in 2×2, 3×3, and 4×4 blocks compression ratios are 75%, 33.33%, and 18.75% respectively. But in the second method, we need to store only two quantities instead of q^2 values in each block. Thus the compression ratios are 50% for 2×2 blocks, 22.22% for 3×3 blocks, and 12.5% for 4×4 blocks. From Figure 2, and Tables 1-3 we can conclude that method 2 shows better CR with reasonable reconstructed image quality measures PSNR and RMSE. The test image Rice achieves an appreciable PSNR 30.3117(dB) with RMSE 7.7796 in the case of 2×2 blocks. As the quality measures PSNR decreases and RMSE increases with an increase of the block size, the algorithm works better with smaller blocks.

Experrimental results shows that PSNR values are greater than 22(dB) and RMSE is less than 20 for different test images when we apply the second method of compression. In practical applications this is an acceptable range at the CR 50%. Also, as the kernel opertion doesn't involve usual convolution product at the edge detection stage the number of additions and multiplications required is reduced and hence saves a lot of computation time.

Test Image	Bolck Size	Block Value	PSNR(dB)	RMSE
Lena	2×2	method 1	22.8412	18.3857
	2×2	method 2	24.8995	14.5066
Cameraman	2×2	method 1	20.7047	23.5127
	2×2	method 2	22.6443	18.8073
Aerial	2×2	method 1	19.9407	25.6747
	2×2	method 2	22.3203	19.5221
Rice	2×2	method 1	27.7290	10.4734
	2×2	method 2	30.3117	7.7796

Table 1: Compression quality measures with 2×2 blocks using methods 1& 2

Test Image	Bolck Size	Block Value	PSNR(dB)	RMSE
Lena	3×3	method 1	16.9428	36.2579
	3×3	method 2	18.0869	31.7830
Cameraman	3×3	method 1	14.9823	45.4389
	3×3	method 2	15.8706	41.0294
Aerial	3×3	method 1	14.3263	49.0033
	3×3	method 2	15.1876	44.3774
Rice	3×3	method 1	21.5605	21.3067
	3×3	method 2	23.6949	16.7801

Table 2: Compression quality measures with 3×3 blocks using methods 1& 2

Test Image	Bolck Size	Block Value	PSNR(dB)	RMSE
Lena	4×4	method 1	12.5169	60.3522
	4×4	method 2	12.8137	58.3253
Cameraman	4×4	method 1	10.5412	75.7666
	4×4	method 2	10.9338	72.4182
Aerial	4×4	method 1	8.7710	92.8943
	4×4	method 2	9.0363	90.0997
Rice	4×4	method 1	16.8793	36.5237
	4×4	method 2	17.5469	33.8217

Table 3: Compression quality measures with 4×4 blocks using methods 1& 2

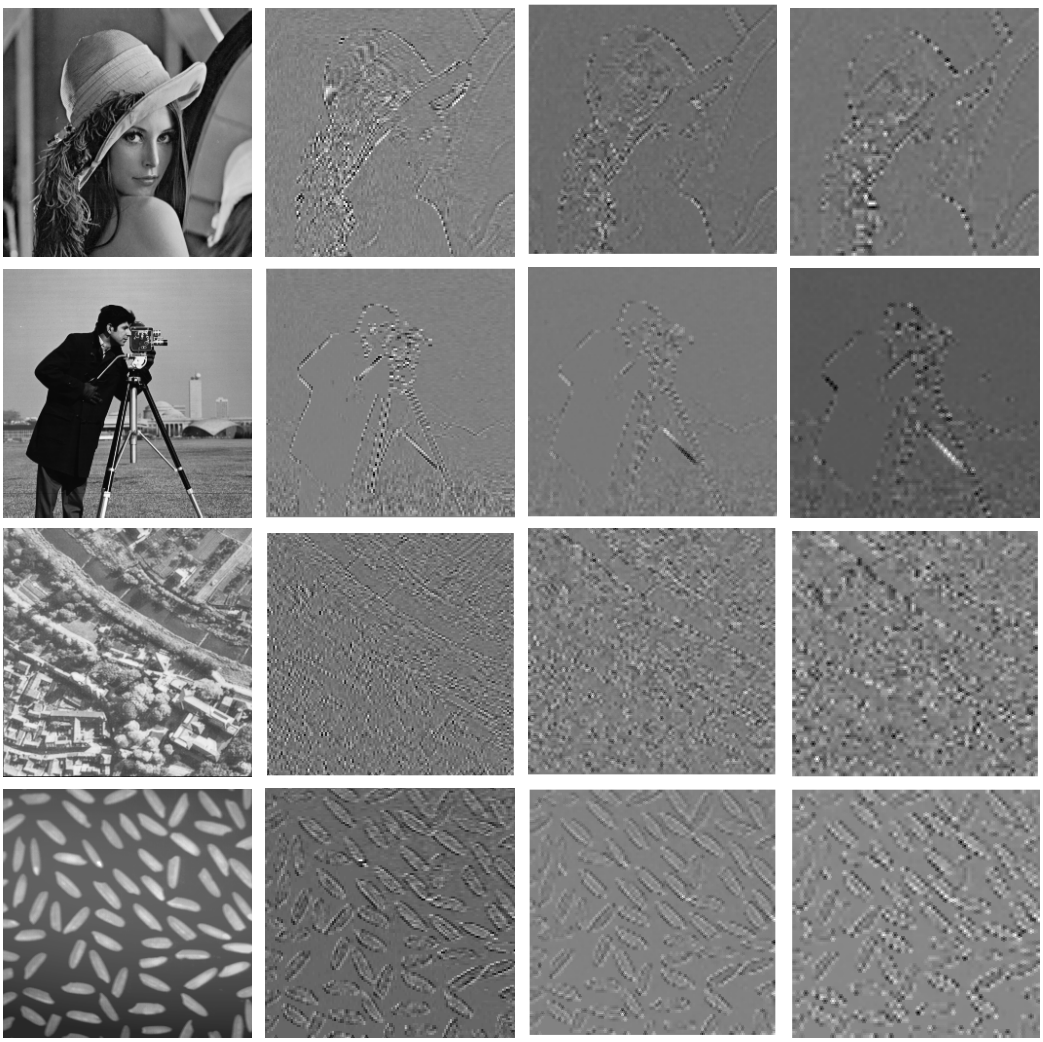

Figure 1: **The first column:** orinal images; **the second column:** edges by M_2; **the third column:** edges by M_3; **the fourth column:** edges by M_4.

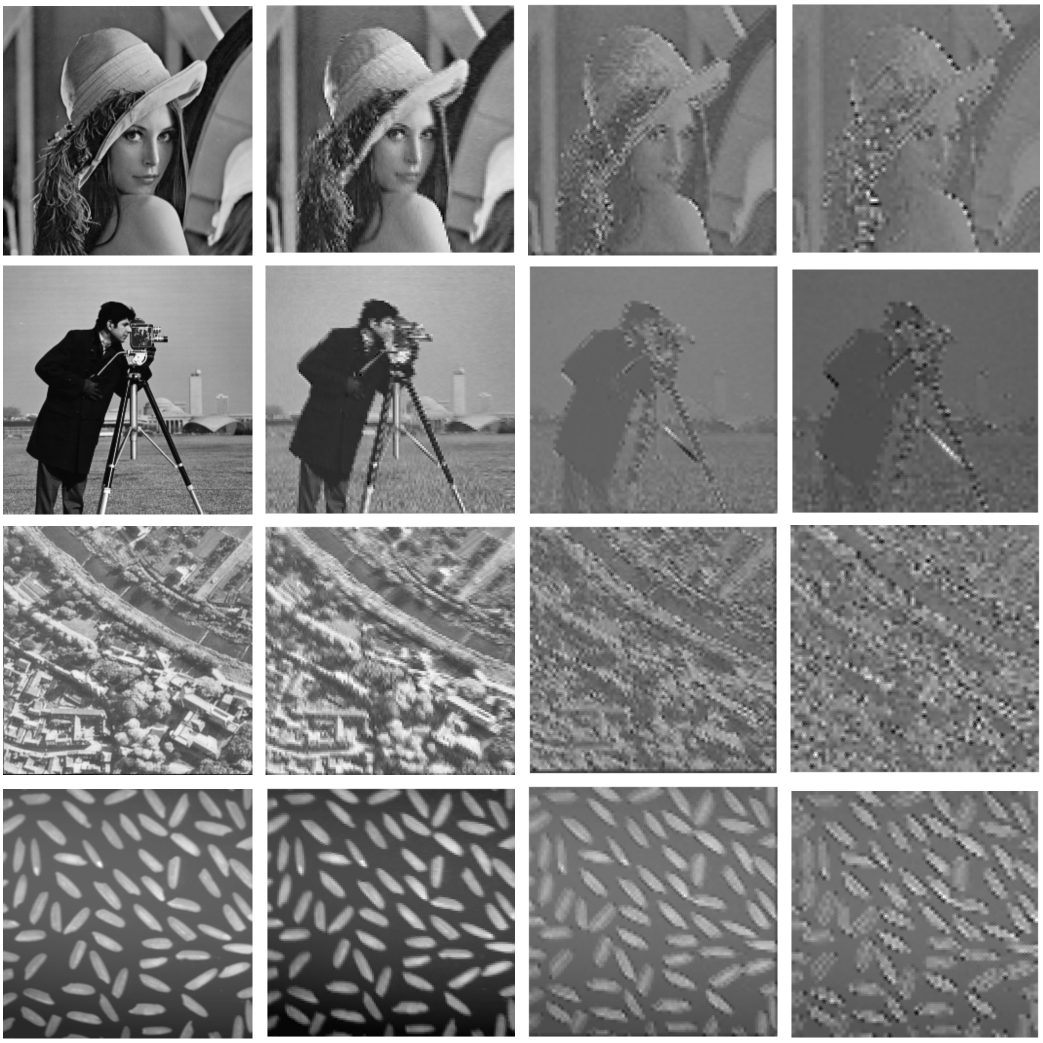

Figure 2: **The first column:** orinal images; **the second column:** reconstructed image using method 2 at the CR 50%; **the third column:** reconstructed image using method 2 at the CR 22.22%; **the fourth column:** reconstructed image using method 2 at the CR 12.5%.

6 Conclusions

In this paper, we presented an edge detection and compression algorithm based on Ramanujan Sums and measures of central tendency and dispersion such as mean and standard deviation. The edge detection algorithm using kernels constructed from Ramanujan Sums has been extended to a compression algorithm. Here the compression is achieved by replacing each block of pixels with a single value obtained by adding edges with texture. The advantage of this method is its low computational complexity and fast reconstruction.

References

Tom M Apostol. Analytic number theory. 1976.

Eckford Cohen. A class of arithmetical functions. *Proceedings of the National Academy of Sciences of the United States of America*, 41(11):939, 1955.

Eckford Cohen. Representations of even functions (mod r), arithmetical identities. *Duke Mathematical Journal*, 25(3):401–421, 1958.

Murray Eden, Michael Unser, and Riccardo Leonardi. Polynomial representation of pictures. *Signal Processing*, 10(4):385–393, 1986.

Robert Gallager. Variations on a theme by huffman. *IEEE Transactions on Information Theory*, 24(6):668–674, 1978.

Rafael C Gonzalez and Richard E Woods. Digital image processing. *Nueva Jersey*, 2008.

Vivek K Goyal. Theoretical foundations of transform coding. *IEEE Signal Processing Magazine*, 18(5):9–21, 2001.

Anil K Jain. *Fundamentals of digital image processing*. Prentice-Hall, Inc., 1989.

Madhuri A Joshi. *Digital image processing: An algorithmic approach*. PHI Learning Pvt. Ltd., 2018.

P Krishnaprasad and Ajeesh Ramanujan. Ramanujan sums based image kernels for computer vision. In *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, pages 835–839. IEEE, 2016.

Christoph H Lampert. *Kernel methods in computer vision*. Now Publishers Inc, 2009.

Francesca Odone, Annalisa Barla, and Alessandro Verri. Building kernels from binary strings for image matching. *IEEE Transactions on Image Processing*, 14 (2):169–180, 2005.

William B Pennebaker and Joan L Mitchell. *JPEG: Still image data compression standard*. Springer Science & Business Media, 1992.

Srinivasa Ramanujan. On certain trigonometrical sums and their applications in the theory of numbers. *Trans. Cambridge Philos. Soc*, 22(13):259–276, 1918.

I Sadeh. Polynomial approximation of images. *Computers & Mathematics with Applications*, 32(5):99–115, 1996.

S Sajikumar and AK Anilkumar. Image compression using chebyshev polynomial surface fit. *Int. J. Pure Appl. Math. Sci*, 10:15–27, 2017.

David Salomon. *A concise introduction to data compression*. Springer Science & Business Media, 2007.

Khalid Sayood. *Introduction to data compression*. Newnes, 2012.

David Taubman and Michael Marcellin. *JPEG2000 Image Compression Fundamentals, Standards and Practice: Image Compression Fundamentals, Standards and Practice*, volume 642. Springer Science & Business Media, 2012.

PP Vaidyanathan. Ramanujan sums in the context of signal processing—part i: Fundamentals. *IEEE transactions on signal processing*, 62(16):4145–4157, 2014.

Ian H Witten, Radford M Neal, and John G Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987.

Haichao Zhang, Jianchao Yang, Yanning Zhang, and Thomas S Huang. Non-local kernel regression for image and video restoration. In *European Conference on Computer Vision*, pages 566–579. Springer, 2010.