Spine Diseases Detection Using Image Processing.

Jyoti Waykule[1], Swaliha Maindergi[2], Prajkta Patil[3], Pramodini Kanase[4]

[1]Assistant Professor of Electronics & Telecommunication Department,SGI, Atigre, Shivaji University,Kolhapur,India.

[2,3,4] Student of B.E. Electronics & Telecommunication Department,SGI, Atigre,Shivaji University, Kolhapur, India.

Abstract— In this project we are going to use HOG(Histogram of Oriented Gradients) and SVM(Support Vector Machine) for boundary regression for biomedical image segmentation. Here we will use MATLAB for computation of HOG and SVM. MATLAB image processing toolbox will be extensively used for reading, processing, visualising and saving the images.

Keywords :- HOG: Histogram Oriented Gradients, SVM: Support Vector Regression, NFS: Neural Foramina Stenosis, CT: Computer Tomography, MRI: Magnetic Resonance Image.

I. INTRODUCTION

Neural Foramina Stenosis(NFS) is narrowing of the small openings between each vertebral in the spine call foramina which nerve roots pass through the nerve roots that exist the spinal column through neural foramina may become compress leading to pain numbness, weakness of the arm hand, leg problem with waking and balance. The risk of neural foramina stenosis increases with age as we age discs in the spine lose height begin dry out. More than 80% people are affect due to NFS. For this NFS segmentation HOG method are used for detecting its proper area and analysis.

MRI and CT imaging are used to display the stenosis for diagnosis and treatment manual segmentation method by physician is to be used for neural foramina image because of size, shape, appearance varation. Segmentation is the process of automatic detection of boundries.

A] HOG THEORY:

HOG is a feature descriptor used to detect object in image processing and computer vision. A feature descriptor is a representation of an image or an image patch that simplifies the image by extracting useful information and throwing away extraneous information

Typically, a feature descriptor converts an image of size width*height*3(channels) to a feature vector/array of length n. In case of HOG feature descriptor ,the input image is a size 64*128*3 and output feature vector is of length 3780.HOG

descriptor can be calculated for other sizes, but above numbers presented in the original paper so one can easily understand the concept with one concrete example.

In the HOG feature descriptor, the distribution (histogram) of directions of gradients (oriented gradients) are used as features. Gradients (x and y derivatives) of an image are useful because the magnitude of gradients is large around edges and corners (regions of abrupt intensity changes) and we know that edges and corners pack in a lot more information about object shape than flat regions.

A. Gradient Calculation :

In HOG feature extractor, the input image is divided into small parts called cells, normally 8x8 pixels as shown in Fig. 2. Then the gradient in both horizontal and vertical direction is calculated for each pixel within the cell.

Simple [-1, 0, 1] and [-1, 0, 1]T gradient filters are applied to the pixel value $f(x, y)$ in order to obtain both $f_x(x, y)$ and f_y (x,y) which are defined as

$$f_x(x, y) = f(x + 1, y) - f(x - 1, y) \text{.....(1)}$$

$$f_y(x, y) = f(x, y + 1) - f(x, y - 1) \text{.....(2)}$$

Once gradients for both x and y direction are calculated, the gradient magnitude $m(x, y)$ and gradient direction $\theta(x, y)$ could be computed as

$$\mathbf{m(x,y)} = \sqrt{f_x^2(x, y) + f_y^2(x, y)} \text{.....(3)}$$

$$\theta(x, y) = \arctan\frac{f_y(x,y)}{f_x(x,y)} \text{.....(4)}$$

In the case of colour image, the gradients at each pixel would be calculated for all colour components separately and the gradient with the largest magnitude would be selected to be used in next steps of feature extraction.

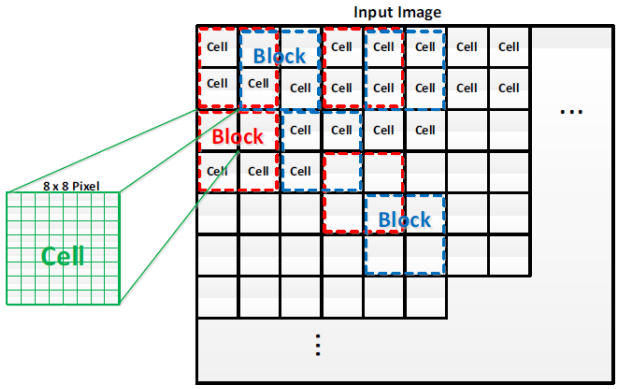

Fig. 2. Cell and block in HOG algorithm. Each cell is defined as an area of 8x8 pixels and each block includes four adjacent cells. Blocks have overlap with their neighbouring blocks.

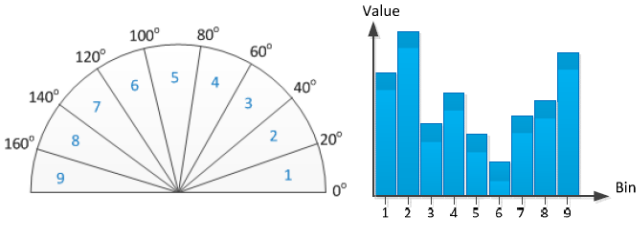

Fig. 3. Orientation bins in HOG and an example of HOG for a cell

B. Histogram Generation

The next step is to generate a histogram of orientation form each cell in the image based on the value of $m(x, y)$ and $\theta(x, y)$ obtained for the pixels within the cell. The interval of $[0,\pi)$ is evenly divided to the number of orientations to produce the orientation bins to be considered in computation of the histogram. The original HOG feature extractor introduced by Dalal and Triggs offers value of 9 to be assumed for number of orientations since it has shown better detection results in the case of human detection [2]. For each pixel within the cell, $\theta(x, y)$ is compared with the direction of all orientation bins to determine which bin it should be associated with. To avoid aliasing effect, two nearest bins to each gradient direction would be updated by defining two different weighting scores based on the distance of gradient angle to the edge angle of each bin [1]. Considering n as number of bins and k as the bin to which $\theta(x, y)$ belongs, percentage of deviation from the bin center, d could be defined as

$$\mathbf{d} = \frac{\left[\theta(x,y) - \frac{k\pi}{n}\right]}{\frac{\pi}{n}} = \frac{n\theta(x,y)}{\pi} - k \text{.....(5)}$$

resulting in $1-d$ as the voting weight of bin k and d as the voting weight for bin $k \pm 1$.

Each gradient would contribute to the histogram by a vote which is basically a function of the gradient magnitude, and could be the square or square root of magnitude, or the magnitude of gradient by itself. It has been shown by Dalal and
Triggs that the best voting parameter is the magnitude of gradient [2]. The votes are also interpolated in position to avoid aliasing, i.e. each pixel gradient would affect its containing cell as well as three adjacent cells by accounting the pixel distance to each cell center. Fig. 3. shows a typical histogram generated for orientation bins within a cell.

C. Block Normalization

The final step in HOG feature extraction is the normalization process across the blocks. Once the gradient for every pixel of the cell is calculated, the adjacent cells are grouped together to form a more spatial block. In fact, blocks are defined as overlapping number of adjacent cells. Normally
each block is comprised of four cells, i.e. a set of 2x2 neighbouring cells [1]. Depending on the normalization scheme adopted for this step feature vectors of the four cells within a block are accumulated to generate a normalization factor. Considering v as the unnormalized feature vector, $\| v \|_k$ as its k-norm, and Ɛ as a small constant, we will have

L1-norm : $\frac{v}{(\|v\|_1+\varepsilon)}$ **……(6)**

L2-norm : $\frac{v}{\sqrt{\|v\|_2^2+\varepsilon^2}}$**……..(7)**

L1-sqrt is defined as L1-norm followed by square root

L1-sqrt : $\sqrt{\frac{v}{(\|v\|_1+\varepsilon)}}$**……..(8)**

and *L2-Hys* is *L2-norm* followed by clipping and limiting the maximum values and then renormalizing [19].

Dalal and Triggs have shown that L2-Hys, L2-norm and L1-sqrt perform equally well for the normalization, while L1-norm reduces the performance by 5% [2]. Omitting the normalization step during the feature extraction has shown 27% decrease in the performance[1,2].

B] Block Diagram :

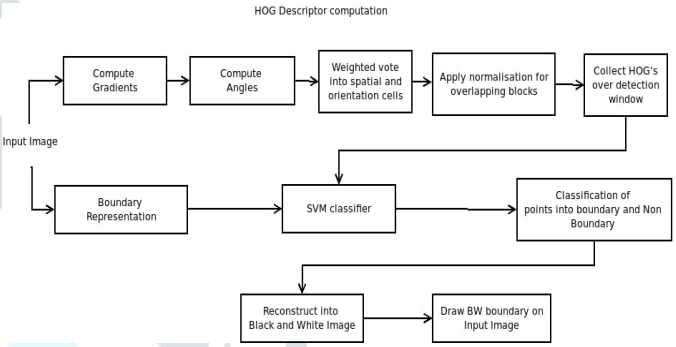

Fig.1. Block diagram

Calculate the Gradient Images:

To calculate a HOG descriptor, we need to first calculate the horizontal and vertical gradients; after all, we want to calculate the histogram of gradients. Gradient of the image is calculated in both "x" and "y" direction (Figure 2.c. and Figure 2.d.). For calculation in x direction 2D filter [-1 0 1] is taken. The filter then calculates the difference between next and previous pixel and stores the result into present pixel as "dx". For calculating gradient in y direction , 2D filter transpose of [-1 0 1] is taken. The transpose forms a vertical array and calculates differrence between vertical pixels and stores reult in "dy".

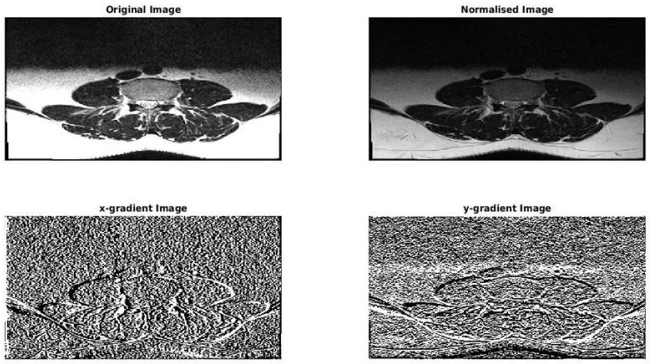

Figure 2. a – d. Output images of each stage

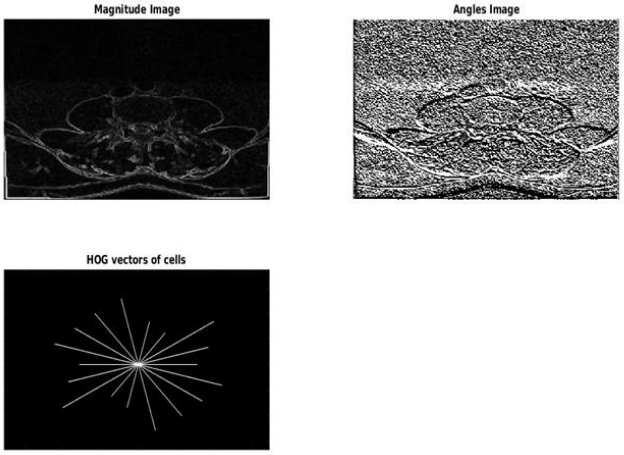

Fig.2.e – g. Output Images of each stage

Then angle (Figure 2.f.) and magnitude (Figure.2.e.) of the gradients are calculates using following two Equations:

Angle=tan-1(dy/dx)

Magnitude=√dx^2+dy^2

The angle and magnitude of gradients is then used for HOG calculations.

Calculate Histogram of Gradients in 8×8 cells:

In HOG feature extractor, the input image is divided into small parts called cells, normally 8x8 pixels. Then the gradient in both horizontal and vertical direction is calculated for each pixel within the cell as given in previous point. Now the values of angles and magnitude in each cell(64 of angles and 64 of magnitudes) are to be formed into histogram of 9 bins making it 9 values for each cell.

The bins on x axis of the histogram are the angles of gradients calculated above. The angles are divided into 9 values ranging from 0 – 180 degrees in steps of 20 degree each. Thus the 9 bins are [0, 20, 40, 60, 80, 100, 120, 140, 160] (180 is not a part of this array as in angular graphs highest will is similar to zero). Then the angles between the bin values are seperated into left portion and right portion. The left portion is contribution to nearest bin value less than the angle and right portion is contribution nearest bin value greater than the angle. The magnitude of the portion is difference between the angle and the bin.

Now the gradient magnitudes corresponding to the bins are added and the histogram is formed (Figure.2.g.).

Block Normalization (Contrast):

Gradients of an image are sensitive to overall lighting. If you make the image darker by dividing all pixel values by 2, the gradient magnitude will change by half, and therefore the histogram values will change by half. Ideally, we want our descriptor to be independent of lighting variations. In other words, we would like to "normalize" the histogram so they are not affected by lighting variations.

Calculate the HOG feature vector:

All the descriptors given by block normalization will be divided into bins according to their contribution forming matrix of size m x n x 9 (where m and n are number of vertical and horizontal cells). The matrix is now converted into single array of cellwise histogram values. And this will give final HOG feature vector.

SVM classifier:

Support Vector Machine(SVM) is supervised machine learning technique that is used for classifying data into two classes (binary classification). SVM classifier is basically a hyperplane which divides the data in two parts. The bins divided by the previous block are classified into two parts (boundary and non-boundary).

SVM training :

We now turn to the problem of training an SVM. The first step is, of course, to choose the nonlinear ϕ-functions that map the input to a higher dimensional space. Often this choice will be informed by the designer's knowledge of the problem domain. In the absense of such information, one might choose to use polynomials, Gaussians or yet other basis functions. The dimensionality of the mapped space can be arbitrarily high (though in practice it may be limited by computational resources).

We begin by recasting the problem of minimizing the magnitude of the weight vector constrained by the separation into an unconstrained problem by the method of Lagrange undetermined multipliers. Thus from Eq. 106 and our goal of minimizing ||a||, we construct the functional

$$\mathbf{L(a,\propto)} = \frac{1}{2} \| \boldsymbol{a} \|^2 - \sum_{k=1}^{n} \boldsymbol{\alpha_k} [\boldsymbol{z_k a^t y_k} - \mathbf{1}] \text{....(1)}$$

and seek to minimize L() with respect to the weight vector a, and maximize it with respect to the undetermined multipliers αk ≥ 0. The last term in Eq. 108 expresses the goal of classifying the points correctly. It can be shown using the so-called Kuhn-Tucker construction (Problem 30) (also associated with Karush whose 1939 thesis addressed the same problem) that this optimization can be reformulated as maximizing

$$\mathbf{L(\propto)} = \sum_{k=1}^{n} \propto_i - \frac{1}{2} \sum_{k,j}^{n} \propto_k \propto_j \boldsymbol{z_k z_j y_j^t y_k} \ldots (2)$$

subject to the constraints

subject to the constraintsary Representation :

Boundary of the image taken for segmentation training is traced using image editing software(GNU Image Manupulation Program here).The traced boundary then divides image into two regions, region inside boundary and region outside boundary. Region outside boundary is filled with Black (Zeros) and region inside boundary is filled with White (Ones). And the image is converted into Boolean.

After converting image into Boolean, the pixels on the boundary are detected. All remaining pixels except the pixels on the boundary are filled with zeros. Image now formed makes labels for SVM classifier. White pixels represent boundary pixels and Black pixels represent non-boundary pixels.

The labels generated are used for training of SVM model used for classification The image is divided into cells.HOG and Gradient magnitudes of each cell are given to SVM model. Corresponding cell is labelled boundary cell if there is boundary pixel in the cell. Otherwise the cell is labelled as non-boundary cell. The cell labels are given to SVM model for training.

Classification into Boundary and Non-Boundary Pixels :

Trained SVM models is given with HOG features of the image along with gradient magnitude features. The SVM model gives output array of labels as boundary and non-boundary cells (Ones for boundary pixels and Zeros for non-boundary pixels.)

Reconstructing Boolean Image from Array of Labels :

Labels given by SVM are stored into an array. The array is of "n*m" length, where n is number of Horizontal cells and m is number of vertical cell in the image. Matrix of corresponding cell sizes (n*m) is created with all zeros.Now the array of labels is iterated with "m" segments of length "n". If boundary pixels is found in the array then corresponding element in the matrix ix made one. Then the matrix is converted into Boolean image.

Draw Segmentation Boundary on Original Image :

Boolean image obtained from SVM output label array is resizes into size of original image. Original image is taken. Pixels in original image corresponding to ones in the boundary Boolean image are marked with red color.

Result and Disscussion

Spine image segmentation methodology explained in this paper is implemented using MATLAB software. Figure 3 shows the results obtained by segmentation of Spine – MRI image using different methods. Figure 3.a. shows segmentation results of HOG – SVM based image segmentation method explained in this paper. The segmentation results are then compared with standard edge detection methods available readily with MATLAB. Prewit image segmentation (Figure 3.b.) gives similar results to our method, but our method removes edges of the image which are out of spine section (advantage of machine learning) and hence is better. Sobel operator (Figure.3.c.) produces same results as of Prewit, with somewhat less fine parts inclusion inside spine image. Canny segmentation (Figure.3.d.) method includes all fine details but also adds high frequency noise.

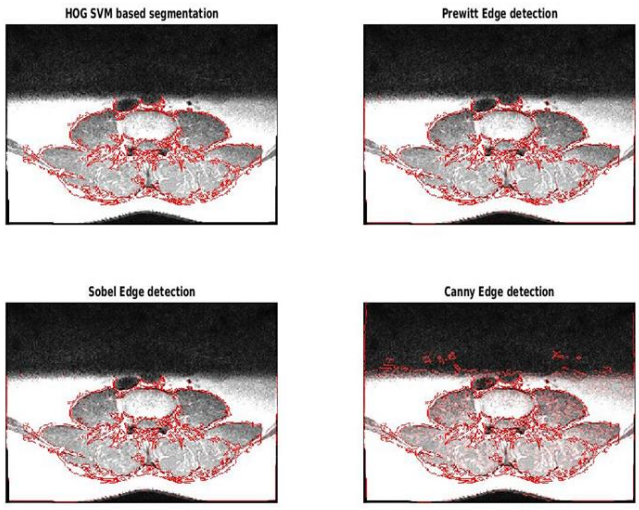

Fig.3. Spine MRI image segmentation using different segmentation methods.

II. CONCLUSION

HOG – SVM based image segmentation produces better results than that of most of the standard image segmentation methods. Unlike standard image segmentation methods, the method requires a large set of training images for SVM to classify more acuurately, hence cannot be operated on new class of images. SVM gives advantage of more accurate and noiseless boundary regression to the method and hence performs better than that of standard image segmentation methods.

The grading system for foramina stenosis of the lumber spine showed nearly perfect inter observed and intra observer argument and would be helpful for study. Segmentation method is important to diagnosis stenosis.

REFERENCES

1.N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., 2005, vol. 1, pp. 886–893.

2. K. Sakthivel, R. Nallusamy, C. Kavitha, “Color Image Segmentation Using SVM Pixel Classification Image”,World Academy of Science, Engineering and Technology International Journal of Computer and Information Engineering Vol:8, No:10, 2014.

3.Canny, John, "A Computational Approach to Edge Detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-8, No. 6, 1986, pp. 679-698.

4.Sobel, I., Feldman, G., "A 3x3 Isotropic Gradient Operator for Image Processing", presented at the Stanford Artificial Intelligence Project (SAIL) in 1968.

5. Prewitt, J.M.S. (1970). "Object Enhancement and Extraction". Picture processing and Psychopictorics. Academic Press.

6.S. Belongie, J. Malik, and J. Puzicha. Matching shapes. The 8th ICCV, Vancouver, Canada, pages 454–461, 2001.

7. V. de Poortere, J.Cant, B. Van den Bosch, J. de Prins, F. Fransens, and L. Van Gool. Efficient pedestrian detection: a test case for svm based categorization.

8. P. Felzenszwalb and D. Huttenlocher. Efficient matching of pictorial structures. CVPR, Hilton Head Island, South Carolina, USA, pages 66–75, 2000.

9. W. T. Freeman and M. Roth. Orientation histograms for hand gesture recognition. Intl. Workshop on Automatic Faceand Gesture-Recognition, IEEE Computer Society, Zurich, Switzerland, pages 296–301, June 1995.

10. W. T. Freeman, K. Tanaka, J. Ohta, and K. Kyuma. Computer vision for computer games. 2nd International Conference on Automatic Face and Gesture Recognition, Killington, VT, USA, pages 100–105, October 1996.

11. D. M. Gavrila. The visual analysis of human movement: A survey. CVIU, 73(1):82–98, 1999.

12. D. M. Gavrila, J. Giebel, and S. Munder. Vision-based pedestrian detection: the protector+ system. Proc. of the IEEE Intelligent Vehicles Symposium, Parma, Italy, 2004.