Title

Abstract Representation of Spatial Data in Machine Vision: A Concise and Flexible Inference Framework

Author information

Nithin PS
Kozhikode, Kerala, India
Email : nithin.pulpra@gmail.com

Abstract

In machine vision, spatial data takes the form of two-dimensional images captured by digital sensors. This study presents an innovative approach to abstractly represent image data, focusing on grayscale images for simplicity. The method involves reducing image contents into concise forms, preserving critical information for efficient and flexible inference without sacrificing the image's signature. Through a series of stages, the boundaries of image blobs are abstracted, resulting in a hierarchical representation. Signatures are then derived from these abstractions, capturing the convexity of abstract points. This framework facilitates machine learning and pattern recognition tasks by enabling the formation of knowledge bases and recognition of novel inputs. Here, we demonstrate how this approach can enhance the analysis and interpretation of spatial data in machine vision applications.

An open source implementation of the methods discussed in the paper is available at https://github.com/ps-nithin/pyrebel-ml

Introduction

This paper proposes a novel method for representing two-dimensional image data into an intermediate form that bridges between raw sensory visual input and human level symbolic cognition.
For the purpose of this study the paper considers grayscale images instead of color images for reducing the complexity of the proposed algorithm / pseudo code.

Existing machine learning methods are predominantly based on neural networks. Neural networks are based on models, trained for achieving specific tasks by using datasets. Datasets required for a training a particular model may not be suitable for training another model and a model trained for a particular task may not perform well for another task. Also, neural networks requires large amount of datasets for training the models and lacks the ability to learn new patterns in realtime. In contrast to neural networks this paper proposes a method that relies on creating a knowledge base using few ideal data sets of the input in real time, thereby drastically reducing the time period between training and inferencing. The fundamental unit of this intermediate representation is called a signature. A signature is a string composed of 1’s and 0’s. The length of the signature gives you the layer at which the signature resides. Shorter the length of the signature, more abstract the signature is whereas longer the signature is, less abstract the signature is. The layer at which a signature resides also defines the abstractness of the signature. The highest level of abstraction is at layer zero and as we move up the layers the level of abstraction decreases and the abstract representation becomes closer to the input data. Every pixel contributes to the signature and hence there is no compromise on leaving out any meaningful inference that can be obtained from the input.

Few terms which will be used from time to time.

1. Pixel

Two dimensional images are stored in memory as arrays. So, if the image has 640 columns and 480 rows , the data would be stored in an array of size 640*480, which equals to 307200. Each item in the array represents a position in the 2D image.

2. Blob

Group of pixels which have the same color and are connected.

3. Boundary of the blob

The pixels around the circumference of a blob.

4. Seed of the blob

The top left pixel in the boundary of a blob.

5. Resolution

The least amount of change in direction that is considered as perceived.

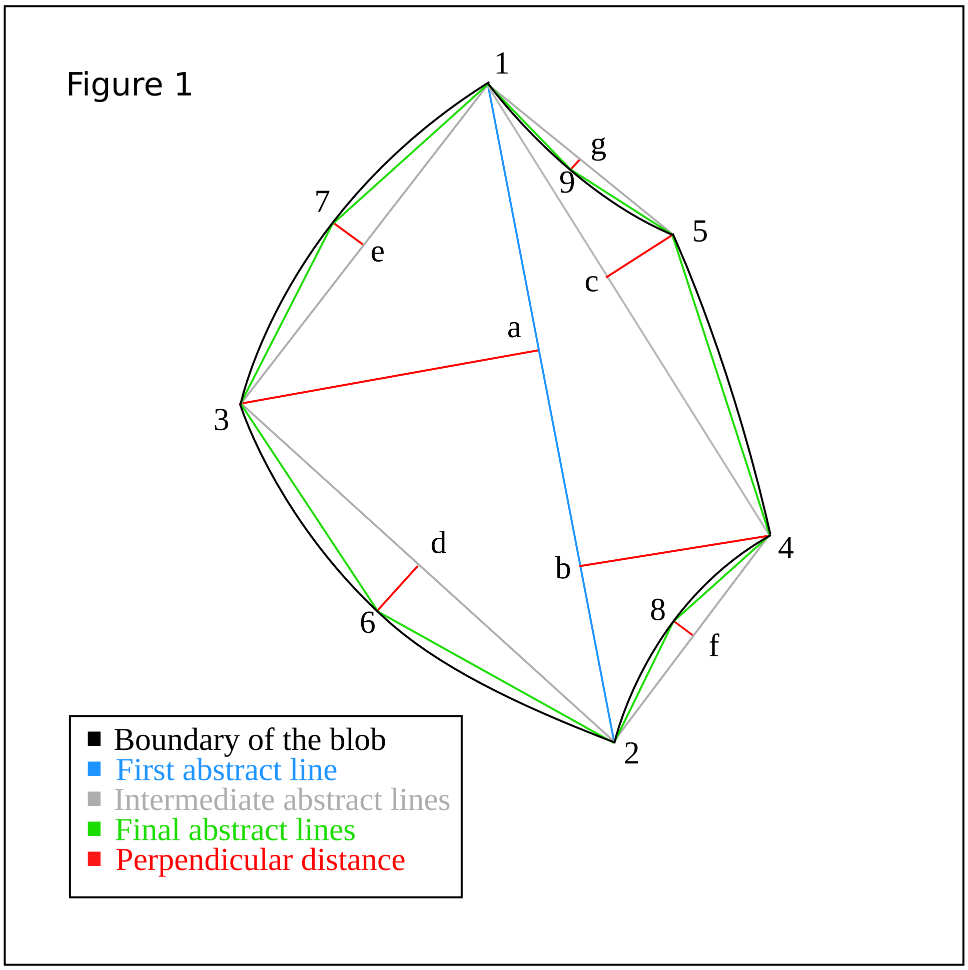
Methods

A 2D image can be considered as a collection of blobs. The first stage of abstraction involves representing blobs in a reduced form. This is called the boundary of the blob ie an ordered collection of pixels around the circumference of the blob.

The second stage involves finding two pixels from the boundary which is farthest from each other. These two points are the first two abstract points of the boundary. The line joining these two pixels is called the first abstract line.

The third stage involves travelling around the boundary and finding a pixel which is farthest from the first abstract line. If the distance of the pixel is greater than the resolution, the pixel is added to the list of abstract pixels. Now we have the first layer of abstraction and the boundary is represented by three abstract pixels and three abstract lines. Similarly, traverse around the boundary of the blob and find the pixel which has the largest perpendicular distance to the corresponding abstract line. If the perpendicular distance is greater than the resolution, the pixel is added to list of abstract pixels and we have a new layer of abstraction.

The process is repeated by travelling around the boundary and finding more abstract pixels till no more abstract pixels can be found for the resolution. As more abstract pixels are obtained the abstract representation becomes closer to the boundary of the blob.



For example in Figure 1,

Length of line segment 1-2 > a-3 > b-4 > c-5 > d-6 > e-7 > f-8 > g-9

Pixels 1 and 2 are the farthest pixels in the boundary and becomes the 1st and 2nd abstract pixels. The blue line joining pixels 1 and 2 is the first abstract line. To find the next abstract pixel traverse around the boundary and find the pixel which has the largest perpendicular distance to the first abstract line. The pixel 3 has the largest perpendicular and becomes the 3rd abstract pixel. The line joining pixel 3 to pixel 1 and 2 becomes the 2nd and 3rd abstract line respectively. To find the next abstract pixel traverse around the boundary and find the pixel which has the largest perpendicular distance to its corresponding abstract line. In the example pixel 4 has the largest perpendicular distance and becomes the 4^{th} abstract pixel. The line joining pixel 4 to pixel 2 and 1 becomes the 4^{th} and 5^{th} abstract line respectively. The process is continued to obtain more abstract pixels till the perpendicular distance becomes less than the threshold or resolution. Now we have an abstract representation of the boundary.

Signatures

A blob may be represented with fewer points with abstract pixels. The signature of the blob is defined as the ordered list of convexity of each abstract point as you traverse the abstract pixels in any direction.
For example in the figure, the list of abstract pixels in clockwise direction starting from abstract pixel 1 would be [1,9,5,4,8,2,6,3,7]. The convexity of an abstract pixel is represented by either 0 or 1. So, in the figure, convexity of abstract pixel 1 would be "1" while the convexity of the next abstract pixel 9 would be "0" and so on. Hence, the signature of the blob at this level of abstraction would be "101101111". A signature are computed for each layer of abstraction.

Layers of signatures

A blob is represented by not a single signature but a series of signatures ranging from the most abstract to the least abstract representation.

Applications

Machine learning and pattern recognition using signatures. The program consists of three parts

1. Forming an abstract representation of data which is used to obtain meaningful information from the input in the form of signatures.

2. Learning. Creating a knowledge base based on signatures obtained in step 1.

3. Recognition. Obtain signatures for novel input and compare it with the signatures stored in knowledge base to recognize the input pattern.

Running the demo program

Learning phase

Patterns and figures have a definite boundary. Each boundary is represented by a signature. Not a single signature but layers of signatures ranging from the most abstract to the least abstract representaion.

Running the following command reads the input file to obtain signatures upto the layers specified and writes it into the knowledge base.

python3 pynvrebel.py --learn <filename.png> or <path/to/learn/> --layer <layers>

For example, running the command

python3 pynvrebel.py --learn images/c.png --layer 20

learns the file c.png and links the signatures with the symbol "c.png".

Similarly, running the command

python3 pynvrebel.py --learn images/letters_standard/ --learn 20

learns all the files in the directory images/letters_standard/ and updates the knowledge base.

Recognition phase

Layers of signatures is obtained for the input data. It is then compared to known signatures in the knowledge base to identify the pattern.

Running the following command reads the input file, obtains signatures upto the layers specified and compares it with the knowledge base to identify any learned patterns. The recognized symbols are then displayed.

python3 pynvrebel.py --recognize <filename.png> --layer <layers>

For example, running the following command

python3 pynvrebel.py --recognize images/c_ripple.png --layer 20

the program checks the signatures in the input image with the knowledge base and displays the recognized symbols, if any.

Statements and declarations

Competing Interests
The author states that there is no conflict of interest.

Funding Information
Not applicable

Author contribution
Not applicable

Data Availability Statement
Not applicable

Research Involving Human and /or Animals
Not applicable

Informed Consent
Not applicable

References

Nil