Small Object Counting with Cellular Neural Networks

Gerhard Seiler

Institute for Network Theory and Circuit Design, Technical University Munich
Arcisstr. 21, D-8000 München 2, FRG

Abstract

This report presents a completely cellular neural network-based system architecture for *small object counting*, where the center positions of small patterns of known shape, size and orientation are located in an input image, in order to be finally counted.
The system consists of three cascaded image processing stages: *Preprocessing* performs noise filtering and contrast enhancement, *pattern matching* approximately locates object positions, and *isolating* ensures uniqueness of perceived object center locations.
Some novel templates for isolating are presented; their stability is proven.

1) Object Counting

Cellular Neural Networks (or CNNs) as introduced in [1] in 1988 are a novel neural network architecture especially suited for analog VLSI implementation and image processing applications. For a general and concise introduction to the field, refer to [7] and [8].

Although CNN research has so far centered on the most fundamental aspects of theory and circuit design, the versatility of the field and the level of rigor in its theoretic development are paving the way to promising applications research. Here, the results of one such project are reported.

Definition 1: Object Counting

Consider an object of known size, shape and orientation, the so-called *target*. Then *object counting* is the task of determining the center positions of all targets present in an input image, and counting them. Other objects that are not to be detected are also allowed to be present. Finally, objects may *collide*: They are allowed to touch or even partially overlap each other. ❑

Fig. 1 illustrates the simple example used throughout this text, where the target is a black disk, whose diameter with respect to the eventual sampling grid is three pixels. To add a bit of realism, contrast in the input image is generally low and not even constant.

The three dots in the ideal output image correspond to the center positions of the isolated target in the top right corner, and the two colliding targets in the bottom left corner. The unwanted other objects, the wavy line and the small disk, should not be detected. Finally, the dots in the output are counted, yielding the number of targets in the input.

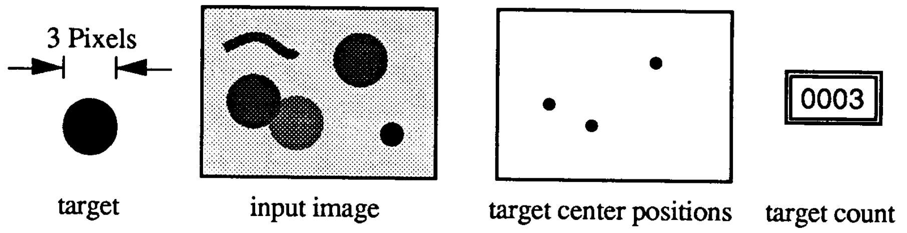


Fig. 1: Object counting example

The object counting problem arises for instance in medical blood screening tests, which require the counting of erythrocytes (red blood cells). Blood also contains eight classes of other objects, which are not to be counted, but can be distinguished from the target erythrocytes by sheer size. Object counting could also serve as part of a more complicated astronomical application for classifying, locating and counting different magnitude stars in sky surveys.

2) System Structure

Fig. 2 shows the basic structure of an *object counting system*. Information flow is from left to right; the boxes represent *subsystems*, the oblique panels *images*. The system produces an *output image* with a single black pixel corresponding to every target present in the input.

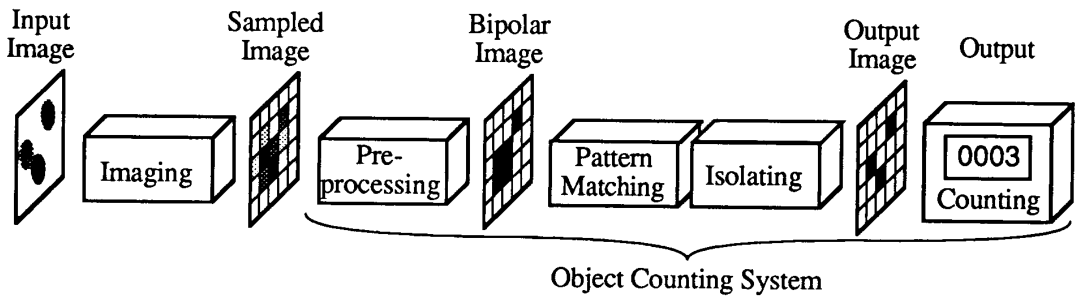


Fig. 2: Basic system structure for object counting

Excepting the trivial task of counting, but including the initial imaging stage (which is not part of the actual object counting system) the individual stages are discussed below.

3) Imaging and Preprocessing

The purpose of both *imaging* and *preprocessing* is to transform the respective input image into a suitable input for the next stage of processing. Specifically, the following image forms are involved:

- The *input image* can be described as a function I: $\mathbb{R}^2 \rightarrow \mathbb{R}_0^+$. It satisfies $I(x, y) \geq 0$ because it is defined as a (positive) physical energy flux density.
- The *sampled image* is a function u: $\mathcal{CG} \rightarrow [-1, +1]$, where $\mathcal{CG}$ is the discrete *cell grid* of the following CNN-based processing stages [7], [8].
- Finally, a *bipolar image* has pixel values of +1 or –1 only, its form is: y: $\mathcal{CG} \rightarrow \{-1, +1\}$.

Most properties of imaging and preprocessing follow directly from these transformation requirements.

3.1) Imaging

Imaging can be modelled as a combination of filtering, sampling and range compression.

First, *low-pass-filtering* occurs, mostly due to integration of the input over some finite sensor area. This is best described as a convolution of the input image with a two-dimensional kernel s(x, y):

$$w(x, y) := I(x, y) \circ s(x, y) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} I(\overline{x}, \overline{y})\, s(x - \overline{x}, y - \overline{y})\, d\overline{y}\, d\overline{x} \qquad (3.1)$$

Sampling then produces a function $\tilde{w}$: $\mathcal{CG} \rightarrow \mathbb{R}$, whose support is restricted to the discrete points of $\mathcal{CG}$. The common square and hexagonal grids are both isomorphic to the integer grid $\mathbb{Z}^2$, such that coordinates can be given in the form (i, j) where $i, j \in \mathbb{Z}$. Using x(i, j) and y(i, j) to denote the coordinates in the real plane $\mathbb{R}^2$ which correspond to the grid point (i, j), sampling can be described as:

$$\tilde{w}(i, j) := w(x(i, j), y(i, j)). \qquad (3.2)$$

This is finally mapped into the allowable range for CNN input by a nonlinear monotonic *range compression* function f_c: $\mathbb{R} \rightarrow [-1, +1]$:

$$u(i, j) := f_c(\tilde{w}(i, j)) \qquad (3.3)$$

Obviously, imaging may profoundly influence the performance of the object counting system.

3.2) Preprocessing

The most obvious choice of a preprocessor is a noise removal CNN, such as the one in Fig. 3 [2]. More sophisticated preprocessors, which automatically and locally adapt to the contrast and brightness of the sampled image, could be developed using a diffusion-based adaptive thresholding mechanism.

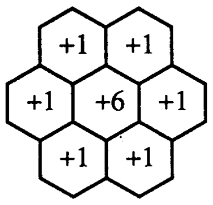
Hexagonal Noise Filter

no u

$i = 0$

$x(0) = \text{image}$

$a =$



no b

Fig. 3: Hexagonal noise filtering CNN parameters[1]

3.3) Imaging and Preprocessing Example

If the targets are small, imaging and preprocessing may lead to *object distortion*: Consider for example a diameter-3 disk as target, and pixel-sized hexagonal sensors. The net effect of imaging then is to integrate the input over the hexagons centered on the sampling positions.

In Fig. 4, the target is well aligned with the sampling grid. Imaging then yields an ideal result: a disk of seven black hexagons, which will be preserved by any reasonable preprocessor.

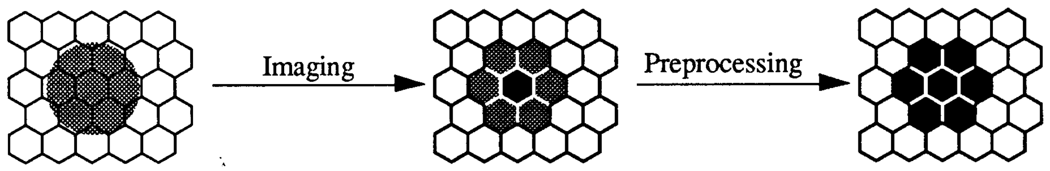


Fig. 4: Imaging and preprocessing a well aligned disk

Things go less well in Fig. 5: The target is badly aligned with the grid; its boundary intersects a lot of hexagons, whose values then are determined by integration to be light or medium grey. The above noise-removal CNN then produces the output shown, a badly distorted target shape.

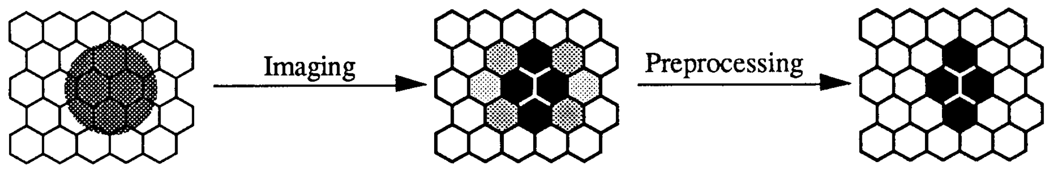


Fig. 5: Imaging and preprocessing a badly aligned disk

1 All CNN parameters are given as normalized quantities as defined in [7].

This phenomenon of shape distortion takes place independent of the detailed choice of the convolution kernel s(x, y), the range compression function f_c and the template used for noise removal.

4) Pattern Matching

Pattern matching is done by performing a spatial correlation with a *pattern function* p(i, j) that resembles the target, and comparing the result with a suitable *threshold* θ:

$$\bar{y}(i, j) := u(i, j) \circ p(i, j) := \sum_k \sum_l u(k, l)\, p(i + k, j + l) \qquad (4.1)$$

$$y(i, j) := \begin{cases} +1 & \text{if } \bar{y}(i, j) > \theta \\ -1 & \text{if } \bar{y}(i, j) < \theta \end{cases} \qquad (4.2)$$

The input to the correlation system must be bipolar in order to provide a constant level of contrast. Only then can a meaningful threshold be specified [3].

As proven in [4] and [7], correlation and thresholding can be done by a single *linear threshold layer*, a CNN whose control template resembles the pattern function p(i, j) to be detected, whose offset equals the negative threshold: i = – θ, and whose feedback template contains only the single self-feedback coefficient which must (as usual) be larger than one.

The parameters of such a linear threshold layer for the detection of disks with a diameter of three pixels are given in Fig. 6:

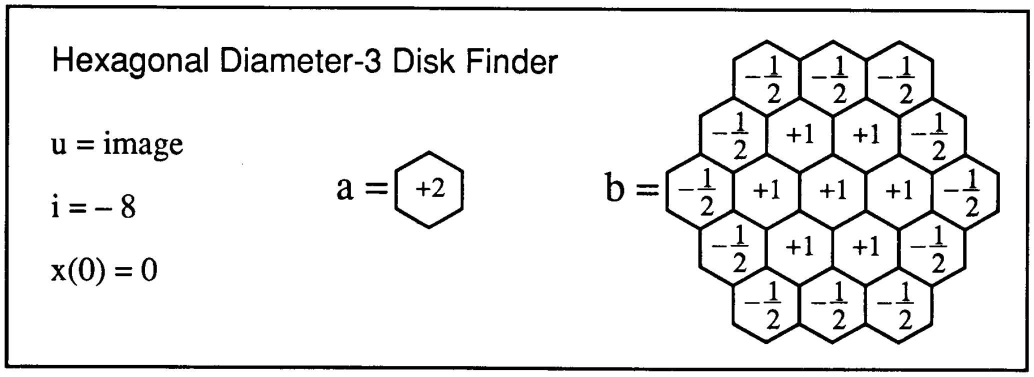


Fig. 6: Hexagonal correlation and thresholding CNN parameters for diameter-3 disks[2]

[2] In practice, a lower offset (a higher threshold) such as i = – 9 would be chosen for this task. However, diameter-3 disk counting then is too simple to allow an illuminating non-trivial example of object counting system operation to be given. The value i = – 8 has therefore been chosen for the sake of the example 5.3.

For every target shape, p(i, j) and θ must of course be carefully designed. Actual development work for a practical application would always have to include numerous simulations, and careful optimization of all parameters involved.

5) Isolating

In general, the output of pattern matching is *blurred*, that is, an above-threshold correlation match often occurs at several adjacent positions. Thus, the output corresponding to a single target may consist of several black pixels, and colliding objects may even lead to a single black zone covering all their respective centers (a particularly bad situation, which is shown in 5.3).

To ensure a one-to-one correspondence between targets and black output pixels, some educated guess at the actual target center positions must therefore be done after pattern matching. A heuristic approach, mimicking what a human observer would do in this situation, is the following:

Definition 2: Isolating

An image processing function performs *(distance) n-Isolating*, where n ≥ 2, if it satisfies the following requirements when applied to a bipolar input image:

- The output is bipolar, and white input pixels always produce white output.
- Any two black output pixels have a mutual distance of at least n pixels.
- The (n–1)-neighbourhood of any black input pixel contains at least one black output pixel.

If n is small, n-isolating may be called *fine-grained*, otherwise it is *coarse*. ❑

This definition does not uniquely specify the action of an isolating system. In fact, it cannot, as isolating is inherently symmetry-breaking: Consider for instance a white image that contains only two adjacent black pixels. Then isolating must yield a single black pixel, corresponding to either one or the other input pixel. This situation is illustrated in Fig. 7.

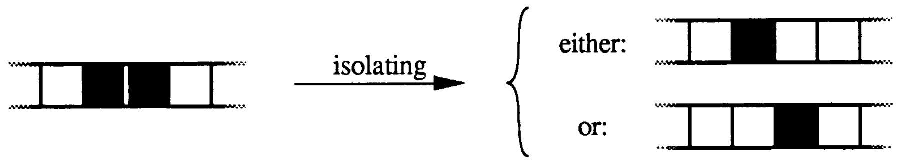


Fig. 7: The action of isolating on two adjacent black pixels

5.1) Linearly Extended 2-Isolating

On the one-dimensional grid, the family of CNNs described in detail in [6] approximates 2-isolating. For most practical purposes, the set of parameters given in Fig. 8 should be adequate:

Linearly Extended 2-Isolating

no u
$i = -1.1 < -1$
$x(0)$ = image

$a =$

–0.9	+2	–1.1

no b

Fig. 8: CNN parameters for linearly extended 2-isolating

This CNN belongs to the family of alternating sign CNNs, whose stability is proven in the appendix. As a noteworthy peculiarity, instead of relying on noise to break symmetry, this CNN deliberately has an asymmetric structure, in order to reduce settling time!

5.2) Hexagonal 2-Isolating

Hexagonal 2-isolating, which is sufficient for diameter-3 disk counting, can be implemented by successively applying linearly extended 2-isolating to the (0, 1)-, (1, –1)- and (–1, 0)-gridsets[3] of the hexagonal plane, as shown in Fig. 9.

Hexagonal 2-Isolating

successively apply:

no u, no b
$i = 0$
$x(0)$ = image

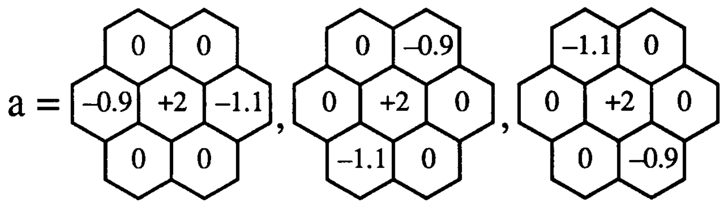


Fig. 9: Hexagonal 2-isolating by successive linearly extended 2-isolating on gridsets

3 A *gridset* is the partitioning of a grid into isomorphic subgrids, on which some CNN action is then performed. For a thorough introduction to the theory of grids for CNNs, refer to [8].

5.3) Pattern Matching and Isolating Example

As a non-trivial example of how isolating complements pattern matching, consider Fig. 10: Preprocessing renders two well aligned target disks into two touching disks, similar to what happened in Fig. 4. The corresponding output of diameter-3 disk finding is a bent four pixel long line, which then is correctly broken up into just two separated black pixels by hexagonal 2-isolating.

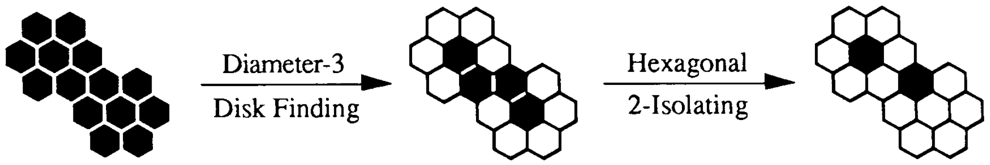


Fig. 10: The action of pattern matching and isolating on two well aligned colliding disks

As can be clearly seen, the center of the lower disk is not detected at the correct position, but shifted by one pixel. This error is comparable in size to the one introduced by sampling, and therefore acceptable.

6) Conclusion

Currently, only fine-grained isolating is practical, which limits the maximum allowable target size and narrows the scope of this paper from object counting in general to *small object counting* in particular.

CNNs are certainly not yet suited for commercial applications design. However, the lines along which such design will eventually be conducted are beginning to emerge, and further applications oriented basic research to fill the future CNN engineer's bag of tricks will be extremely worthwhile to pursue.

At the time of this writing, the author intends to specifically investigate planar and higher order isolating based on the reported and other templates, and related custom nonlinear systems.

Acknowledgement

This report summarizes the work the author has done as a visiting researcher at UC Berkeley in early 1990. He wishes to thank Prof. Leon O. Chua and Prof. Josef A. Nossek for making this research stay possible, and for numerous helpful comments during the course of this work.

Appendix: Stability of Alternating Sign CNNs

An easy proof of stability has been found for yet another class of linearly extended CNNs, which encompasses the family discussed in [6], and particularly the 2-isolating CNN used in this text:

Definition 3: Alternating Sign Templates

A template a = [...a(–2) a(–1) a(0) a(1) a(2)...] is called an *alternating sign template*, if it both satisfies the *sign condition* $a(i)\begin{cases}\geq 0 & \text{if i is even}\\ \leq 0 & \text{if i is odd}\end{cases}$ and the (stricter) *value condition* $a(i)\begin{cases}>1 & \text{if } i=0\\ <0 & \text{if } i=\pm 1\end{cases}$. ❑

Theorem 1: Stability of Alternating Sign CNNs

Any linearly extended CNN, whose feedback template is of the alternating sign type, is stable.

Proof: Consider a linearly extended alternating sign CNN without input, and choose $\mathcal{CG} \cong \mathbb{Z}$:

$$\frac{dx^c(\tau)}{d\tau} = -x^c(\tau) + i^c + \sum_i a(i)\, y^{c+i}(\tau) \quad \wedge \quad y^c(\tau) := f(x^c(\tau)) \qquad \text{(A.1)}$$

The notational conventions of [7] have been used here. Then the substitutions

$$\bar{\xi}^c := \begin{cases}+\xi^c & \text{if c is even}\\ -\xi^c & \text{if c is odd}\end{cases} \qquad \text{(A.2)}$$

where ξ^c represents all of x^c, y^c and i^c, and

$$\bar{a}(i) := \begin{cases}+a(i) & \text{if i is even}\\ -a(i) & \text{if i is odd}\end{cases} \qquad \text{(A.3)}$$

define a new system, which is also a cellular neural network:

$$\frac{d\bar{x}^c(\tau)}{d\tau} = -\bar{x}^c(\tau) + \bar{i}^c + \sum_i \bar{a}(i)\, \bar{y}^{c+i}(\tau) \quad \wedge \quad \bar{y}^c(\tau) := f(\bar{x}^c(\tau)) \qquad \text{(A.4)}$$

The sign condition ensures that the new feedback template $\bar{a}$ as defined by (A.3) is positive; and due to the value condition it is cell linking. Then [5], theorem 1 guarantees the stability of (A.4)[4]. As (A.2) establishes a homeomorphism between x and $\bar{x}$, stability of (A.1) follows.

The general situation where input is present is covered by similar additional substitutions of the input and the control template. ❑

4 In the theorem, only homogeneous CNNs are considered (with constant parameters in every cell). However, this requirement is neither needed nor exploited in its proof. [7] gives a general introduction to non-homogeneous CNNs.

References

[1] L. O. Chua and L. Yang, "Cellular Neural Networks: Theory", IEEE Tr. CAS, Vol. 35, pp. 1257 – 1272, 1988

[2] L. O. Chua and L. Yang, "Cellular Neural Networks: Applications", ibid., pp. 1273 – 1290

[3] R. C. Gonzalez and P. Wintz, "Digital Image Processing", Addison Wesley, Reading, 1987

[4] L. O. Chua and B. E. Shi, "Exploiting Cellular Automata in the Design of Cellular Neural Networks for Binary Image Processing", ERL memo UCB/ERL M89/130, University of California, Berkeley, November 15, 1989

[5] L. O. Chua and T. Roska, "Stability of a Class of Nonreciprocal Cellular Neural Networks", ERL memo UCB/ERL M89/100, University of California, Berkeley, September 5, 1989

[6] G. Seiler, "A Cascade of Bifurcations in a Family of Cellular Neural Networks", report TUM–LNS–TR–90–5, Technische Universität München, August 1990

[7] J. A. Nossek, G. Seiler, T. Roska and L. O. Chua, "Cellular Neural Networks: Theory and Circuit Design", report TUM–LNS–TR–90–7, Technische Universität München, October 1990

[8] J. A. Nossek, G. Seiler, T. Roska and L. O. Chua, "Cellular Neural Networks: Topology, Symmetry and Layout", report TUM–LNS–TR–90–8, Technische Universität München, in preparation