Distributed Dynamic Processing for Edge Detection

H. Tunley
School of Cognitive and Computing Sciences, University of Sussex
Brighton, England

Abstract

This paper discusses a dynamic method of edge detection which works from a sequence of frames. Most edge detection algorithms process image information statically, regardless of whether the application is static – i.e. whether the input is a singular, unique image – or a dynamic sequence of frames used for tracking or optic flow extraction. As many applications are dynamic, such as robotics, autonomous vehicle control and satellite tracking, it makes sense for edge detection processes to exploit dynamic phenomena.

Employing dynamic processing offers a number of advantages, the main one being noise reduction. This paper discusses a dynamic edge detection process implemented as a network of simple processing units. In addition to edge detection, the network simultaneously determines optic flow and areas of occlusion and disocclusion. These areas provide information on how the image view is changing, which is useful for such applications as autonomous vehicle guidance.The integration of these two processes helps overcome problems such as feature matching. This paper describes mainly the edge detection process. Details of the optic flow processing have been described elsewhere ([3, 2, 4]).

Following a description of the dynamic processing network, the results of this method are compared to the static edge detection scheme of Canny [1]. The network proves to be an efficient means of obtaining moving edges when a sequence of frames are available. Finally a recent extension for determining edges in images with multiple motions present is outlined.

The Advantages of Dynamic Processing

Noise Reduction

The dynamic method used in this network is very robust to noise distortion. Conventional static methods determine edges at all intensity gradients with no notion as to whether a given 'feature' has permanance. The network described here extracts edges by exploiting the fact that noise does not normally correlate between frames – i.e. noise at one point in an image is unlikely to be present at the same point in the next image (or at a neighbouring point when considering motion). Whilst most correlation methods correlate statically-obtained edge information in the spatial domain, this network uses temporal correlation, both to reduce noise *and* to achieve determine motion. Movement (through occlusion) helps to guide edge detection and vice versa.

If a certain percentage of a given image frame is statistically identified as noise, then the probability of noise correlation between frames can be determined. Firstly the probability of noise at a point (i, j) in a single frame is:

$$P(N_{ijt}) \equiv N_d \tag{1}$$

where N_{ijt} is the noise at (i, j) at time t and N_d is the percentage noise density. Given that after each time interval Δt, a new frame is introduced, the probability of noise being correlatable at the same point is:

$$P(CN_{ij(t+\Delta t)}) = P(N_{ijt}).P(N_{ij(t+\Delta t)}) \tag{2}$$

where $P(CN)$ is the correlation probability. As the noise probability for any single independent frame is the same, i.e. $P(N_{ijt}) = P(N_{ij(t+\Delta t)})$, this can be generalised for any sequence of F frames to give:

$$P(CN_F) = (P(N))^F = {N_d}^F \tag{3}$$

Therefore, if information is integrated over a number of frames, the probability of noise being correlated, and therefore interpreted as valid edge information, is reduced exponentially with frame number.

The Model

The network, shown diagrammatically below, is a heterarchical network of simple processing units. It has an interwoven structure which allows a two-way flow of information, thus motion and feature extraction occur naturally in parallel.

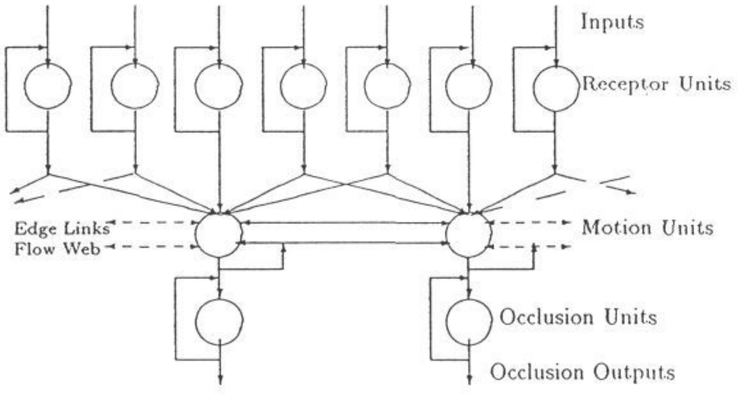


Figure 1: Basic Network Structure

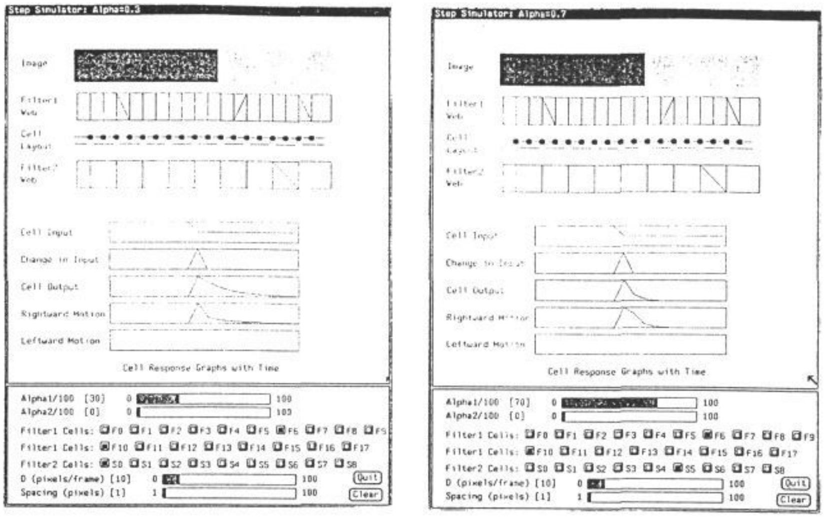
Temporal Processing

A group of input units are used in this network to detect areas of change in input pixel intensity between frames. The response characteristics are determined using an autoregressive filter with the following function:

$$R_{ijt} = (1 - \alpha).R_{ij(t-1)} - \alpha.[I_{ijt} - I_{ij(t-1)}, 0]^+ \tag{4}$$

where R_{ijt} is the filter response at point (i, j) and time t, I_{ijt} is the input image intensity under the same conditions, $[a, b]^+$ is the maximum of a and b, and $\alpha \in [0, 1]$ controls the degree of influence of the previous response.

When changes occur at the input of a receptor unit, R, it responds with a signal 'peak' followed by a decay to a new steady-state value. The unit's output is active whenever changes are occurring (and for a time afterwards, dictated by α), due to the recurrent feedback present in the autoregressive filter. Below are the outputs for a single R cell from a simulation of the movement (rightwards) of a simple step edge for two different values of α (0.3 and 0.7). The simulation also shows the response of the motion cells signalling leftward and rightward motion. As can be seen there is no leftwards activity.



```
Alpha1/100  [30]      0 [        ] 100
Alpha2/100  [0]       0 [        ] 100
Filter1 Cells: F0 F1 F2 F3 F4 F5 [✓ F6] F7 F8 F9
Filter1 Cells: [✓ F10] F11 F12 F13 F14 F15 F16 F17
Filter2 Cells: [✓ S0] S1 S2 S3 S4 S5 S6 S7 S8
D (pixels/frame) [10]   0 [        ] 100
Spacing (pixels) [1]    1 [        ] 100
```

```
Alpha1/100  [70]      0 [        ] 100
Alpha2/100  [0]       0 [        ] 100
Filter1 Cells: F0 F1 F2 F3 F4 F5 [✓ F6] F7 F8 F9
Filter1 Cells: [✓ F10] F11 F12 F13 F14 F15 F16 F17
Filter2 Cells: S0 S1 S2 S3 S4 [✓ S5] S6 S7 S8
D (pixels/frame) [10]   0 [        ] 100
Spacing (pixels) [1]    1 [        ] 100
```

Figure 2: Receptor Output for 1D Image Sequence

Discontinuities in unit activity provide information on the positions of 'leading edges' in a scene – the current positions of the edges of coherent moving regions which can be interpreted as potential featural edges. The amount of activity at a given time depends upon the displacement between frames and upon the speed of adaption – a parameter of the input units. Figure 3 shows the initial responses of the receptor units ($\alpha = 0.7$ and $\alpha = 0.3$) after the first two frames from a sequence of a rotating plug.

Spatial Processing: Finding 'Leading Edges'

Creating a Flow Web

The flow web is a *structure* which spreads receptor activity within a region of units. The web links units capable of detecting edges with those which determine motion. The two processes are so interlinked within this network that it is impossible not to discuss *some* aspects of the motion processing stage.

The flow web consists of *clusters* of edge and motion units (with differing

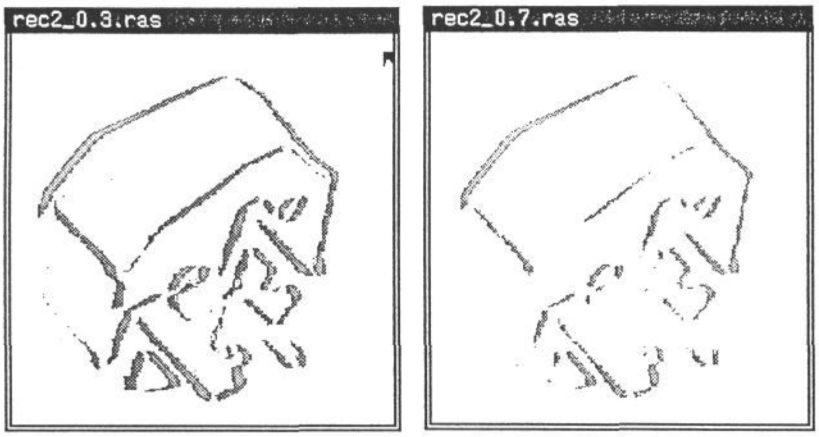


Figure 3: Receptor Output for 2D Image Sequence

sensitivities to orientation and velocity) arranged on a hexagonal grid so that *all* clusters are equidistant. Known as vector units, due to their directional sensitivity, each unit in each cluster is linked to the neighbouring cluster in the direction of its sensitivity. It is through this hexagonal linking structure that the gating actions occur which control both edge detection and flow extraction.

The unit sampling is not at a pixel scale (it is not necessary to determine the flow field everywhere in the image, although relatively close sampling is necessary for accurate edge finding). If the link between two vector units, from (a, b) in the direction k at time t is represented as L_{abkt}, then:

$$L_{abkt} = L_{abk(t-\Delta t)} \sum_{x=a}^{a_1} \sum_{y=b}^{b_1} R_{xyt}. \tag{5}$$

where R_{xyt} is the input receptor unit at (x, y), (a_1, b_1) is the position of the neighbouring vector unit in the direction k, and the summation is pixel-wise along the direction of the link.

Lateral Interaction

As well as receptor unit activation, the lateral connections from neighbouring links influence which motion and edge units produce an output. There are three main interaction rules, but only the one directly relevant to edge detection is discussed here (see [2] for more details):

Vector units with no current input activity cannot influence neighbouring units. This rule is used to isolate 'leading edges'. To explain this further we need to consider the notion of *gating* used to extract optic flow and moving edges.

Gating: Edge Activation for Static Backgrounds

Gating is the process of unit activation due to the activity present in connected units. For this network gating occurs when the activity levels of the links between two motion units are different – i.e. an activity gradient exists. In this situation a flow vector response for the motion unit with sensitivity in the direction of the link's orientation is created with an amplitude proportional to the gradient. Activation thus occurs amongst motion units in the direction of decreasing input activity. Using the labelling strategy established earlier, the two links into a unit at position (a, b) are given by, $L_{a_1b_1kt}$ and $L_{a_2b_2k't}$ where (a_1, b_1) is the position of the cluster *before* the motion unit (in the direction k), and (a_2, b_2) cluster position *after* in the opposite direction k' (i.e. *back* to the unit).

For gating to occur an activity gradient must exist between $L_{a_1b_1kt}$ and $L_{a_2b_2k't}$ and R_{abt} must be active. These constraints are satisfied by:

$$M_{abkt} = R_{abt}.[L_{a_1b_1kt} - L_{a_2b_2k't}, 0]^+ \tag{6}$$

where M_{abkt} represents the motion unit output.

If gating is confined only to those units with zero potential on one side, then vector units will only be activated at the discontinuities of movement areas, some of which will be leading edges. The edge links are activated directly by gated motion units which, in turn, respond to motion discontinuities that could represent moving edges.

Reducing Ambiguity: The Advantage of a Multiple Adaption Rate

Having multiple adaption rates (i.e. variations in α) amongst the receptor units can help in overcoming a potential ambiguity which occurs when considering *all* discontinuities as moving edges (due to the fact that adaption occurs over a period of time, so not just recent movement might be interpreted as edge movement).

Representing fast adapting R units as F units, and slower ones as S, then the following can be stated:

1. At a given position (a, b) if S_{ab} is **active** and F_{ab} is **inactive**, then motion causing the adaption process is **not recent**.

2. If S_{ab} is **inactive** and F_{ab} is **active**, then the motion causing the adaption process is **very recent** and positioned very near to (a, b).

3. If both S_{ab} and F_{ab} are **active** then motion is **recent**, though the positioning of the edge might be offset slightly depending upon the activity of neighbouring F units.

In the last two cases edge units can be activated with a reduced possibility of ambiguity.

Activating Edge Units

Initially all units within a potential leading edge cluster (as defined by the above criteria) are activated to the same degree (proportional to the motion information present). If the edge link at (a, b) with *orientation* k is represented as E_{abkt}, then:

$$E_{abkt} \; \forall \; k = max(M_{abkt}) \tag{7}$$

where $max(x)$ implies the most active unit within the unit cluster x.

Active units are linked into coherent structures by integrating activity between units using three main interaction rules:

1. Any incorrect contour-parallel motion activity can be suppressed by suppressing motion activity 'along' leading edges.

$$\text{If } E_{a_1 b_1 kt} + E_{a_2 b_2 k't} \neq 0 \text{ then } M_{abkt} \rightarrow 0.$$

where (a_1, b_1) and (a_2, b_2) are the positions of neighbouring units each side of (a, b) along a link with orientation k. This suppression is important in that spurious flow web activity will build up strongly *along* straight edges creating strongly-gated motion at the corners of such edges.

2. Any active edge link between motion units with common motion components, should be enhanced.

$$E_{a_1 b_1 kt} = E_{abkt} + \sum_{d=1}^{k} M_{abkt} . M_{a_1 b_1 kt} \tag{8}$$

The $\sum$ is a measure of the similarity in motion vector distribution between clusters at (a, b) and (a_1, b_1); the closer the distribution, the stronger the link.

3. Each motion vector unit in each cluster suppresses its opposing unit to a degree proportional to its own activity.

This ensures that neighbouring motion units with no common motion components suppress active edge links.

$$M_{abkt} = [M_{abkt} - M_{abk't}, 0]^{+} \tag{9}$$

Results

The edge unit output of this network is shown in Figure 4 for comparison with a popular static segmentation method, that of Canny [1] [1]. The Canny algorithm was tried with a number of standard deviations and the best result used for the comparison. The result for the network is after three frames – i.e. the Canny was applied to the third frame in a sequence of a moving object. As can be seen, the network produces far fewer spurious (noise-induced) edges.

[1] Thanks to Alistair Bray for supplying the Canny image.

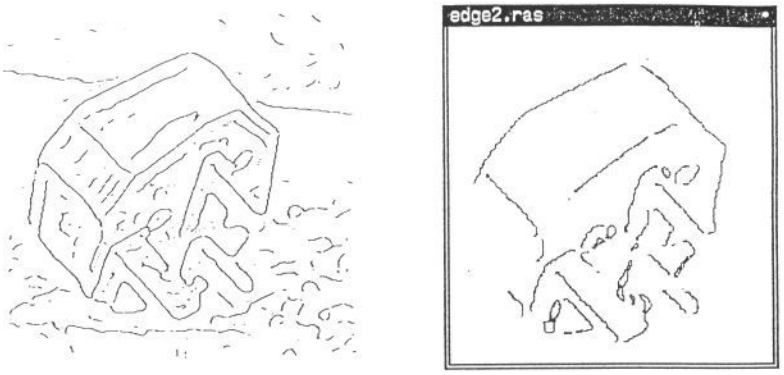


Figure 4: Edge Detection Comparison: Canny and this Model

Extensions to this Method

The method described in this paper is designed to determine edges in images in which the background is static. Recent changes have been made to produce a model capable of segmenting regions with differing movements in a sequence with more than one moving object.

The main theory behind this extension is to detect the *coherence* of areas of image motion for a given sensitivity of standard comparator motion detector. Motion sensitivity is controlled temporally by the filter variable α, and spatially by the distance between motion detector inputs, d:

$$M_{ijkt} = |R_{i'j'kt} - R_{ijk(t-1)}| \tag{10}$$

where M_{ijkt} is the motion cell activity at point (i, j) in the direction k, and i', j' is the cell positioned a predetermined distance from (i, j) in the direction k.

The flow web now consists of *clusters* of motion units (with differing sensitivities to velocity) arranged on a hexagonal grid so that *all* clusters are equidistant. Known as vector units, due to their directional sensitivity, each unit in each cluster is linked to the neighbouring cluster in the direction of its sensitivity. It is through this hexagonal linking structure that the gating actions occur which control edge detection, occlusion and flow extraction.

Spatial Processing: Determining Motion Energy

Flow web activity represents the spatial coherence of motion detector activity. Edge detection involves the determination of changes in coherence with time through the notion of occlusion. The flow unit sampling is not at a pixel scale (it is not necessary to determine the flow field everywhere in the image, although relatively close sampling is necessary for accurate edge finding). A measure of coherence is obtained at each flow position (a, b) using the following measure:

$$F_{abkt} = \frac{M_{abkt}}{\sum_{k=1}^{k=k_{max}} M_{abkt}} \tag{11}$$

where F_{abkt} is the flow web value at sample point (a, b) and each direction in each flow web cluster is represented as an integer between 1 and k_{max} – which for current implementations is 6.

Temporal Processing: Determining Edges

The theory used to achieve dynamic segmentation is based upon the Gestalt notion of *common fate* – areas with similar motion characteristics are statistically likely to belong to the same object. Therefore areas of image which are spatially close, and have similar coherency measures (as defined by F_{abkt} above) can be segmented into a single region of movement.

Finding Occlusion and/or Disocclusion

Edges are deemed to exist at *temporal changes* in motion coherence – i.e. at points where one surface is somehow occluding or disoccluding another. At each flow web position exists, as well as a cluster of flow/coherency cells, a cluster of dis/occlusion detecting cells:

$$O_{abkt} = F_{abkt} - F_{abk(t-\Delta t)} \tag{12}$$

where O_{abkt} is the dis/occlusion at (a, b) in direction k.

The use of the concept of occlusion is useful in that it allows for more than one form of movement to be detected within an image at any one time. For example, the movement of an object in one direction along with the simultaneous movement of the background in another.

Determining Dynamic Edges

Edges are assumed to exist in regions of dis/occlusion, in the minimum direction of occlusion:

$$E_{abkt} = \sum_{k=1}^{k=k_{max}} [O_{abkt}, 0]^{+} \tag{13}$$

where E_{abkt} is the edge activity at point (a, b) and k is the orientation of the detected edge segment.

Conclusions

To conclude, this network achieves edge detection (and the extension determines areas of occlusion and disocclusion) in an efficient manner. The images contain less noise than those obtained by static means and the method would be efficient for any dynamic applications where only moving edges require detection. The distributed nature of the network would also allow for a parallel implementation, making real-time frame-rate processing a definite possibility.

References

[1] J. Canny. A computational approach to edge detection. *IEEE P.A.M.I*, 8, 1986.

[2] H. Tunley. A neural network model for dynamic image processing. *C.S.R.P. 166, School of Cognitive and Computing Sciences, University of Sussex*, 166, 1990.

[3] H. Tunley. Segmenting moving images. In *Proc. Int. Neural Networks Conference (INNC90), Paris, France*, 1990.

[4] H. Tunley. Dynamic image segmentation and optic flow extraction. In *Proc. Int. Joint Conference on Neural Networks (IJCNN91), Seattle, USA*, 1991.