Second Order CNN Arrays for Estimation of Time-to-Contact

Bertram Shi

Department of Electrical and Electronic Engineering
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, HONG KONG
email: eebert@ee.ust.hk

ABSTRACT: This paper describes a cellular neural network (CNN) for estimating the time-to-contact from a one dimensional image. The CNN arrays used for this algorithm consist of cells with second order dynamics. The key feature of these arrays is that the spatial information in a region around each cell is represented by the phase of a complex number. The velocity is encoded as the temporal variation of that phase. By modelling this variation using adaptive temporal oscillators, the velocity can be estimated. Velocity information extracted over the entire array can be combined to estimate time-to-contact.

1. Introduction

When a camera images its environment, each point on the image plane corresponds to a point in the environment. Assuming the camera is translating with constant velocity, the *time-to-contact*[1] of an image point is the time at which the camera will reach the corresponding point in the environment. This representation of the environment is useful for such tasks as obstacle avoidance and autonomous navigation[2].

In this paper, we describe a CNN[3][4][5] array which can estimate time to contact from an image assuming that the camera is approaching a planar object oriented parallel to the image plane. Under this assumption, the time to contact is the same for every image point. Section 2 demonstrates that assuming perspective projection, the image velocity for 1-D image is a linear function of the image position. The slope of the velocity is inversely proportional to the time-to-contact. Section 3 describes how a CNN array with second order cells can perform spatio-temporal image filtering and extract velocity information directly from an image. Section 4 describes a simple circuit which combines the velocity information extracted by the CNN described in Section 3 to estimate the slope of the velocity variation across the image and hence the time-to-contact. Section 5 illustrates the algorithm with the results of computer simulations.

2. Time to Contact with Planar Objects

Assume a one dimensional camera is imaging a two dimensional environment. Let (R, Z) be the environmental coordinate frame and let r be the image coordinates. By suitable choice of the axes, we can choose the image axis r to be parallel to the R axis. Under the assumption of perspective projection[6] (Figure 1), an object at position (R_0, Z_0) in the environment is imaged to the point r_0 in the image plane where

$$r_0 = f \cdot \frac{R_0}{Z_0} \quad (1)$$

and f is the focal length of the camera.

Figure 1: The geometry of perspective projection.

Figure 2: (a) The impulse responses of the even (a) and odd (b) spatial filters for $\lambda = 0.15$ and $\Omega_x = 0.93$.

Assume that there is an planar object parallel to the R axis and located a distance Z_o from the camera. If the camera approaches the object straight on with velocity

$$v = \frac{dZ_o}{dt}. \quad (2)$$

and the object does not deform ($dR_o/dt = 0$), the camera motion induces motion in the image plane given by

$$\frac{dr_o}{dt} = f \cdot \frac{R_o}{Z_o^2} \cdot \frac{dZ_o}{dt}.$$

Substituting equations (1) and (2),

$$\frac{dr_o}{dt} = r_o \cdot \frac{v}{Z_o} = \frac{r_o}{t_c} \quad (3)$$

where $t_c = Z_o/v$ is the time-to-contact. Thus, the velocity of an image point (dr_o/dt) increases linearly with its position (r_o). The slope is inversely proportional to the time to contact.

3. Spatio-temporal Image Filtering

In this section, we describe a second order CNN array for spatio-temporal filtering of an one dimensional image. The filtering is velocity specific, i.e. the filters are tuned to respond to image motion at a given speed and direction. The outputs of these filters give information about the local image velocity around each pixel.

The filtering is separable in space and time. In other words, we can consider the filter to be the cascade of a spatial filtering stage and a temporal filtering stage. In the spatial filtering stage, the image is convolved with two convolution kernels approximately equal to

$$h_e(n) = \frac{\lambda}{2} e^{-\lambda|n|} \cos(\Omega_x n) \text{ and } h_o(n) = \frac{\lambda}{2} e^{-\lambda|n|} \sin(\Omega_x n). \quad (4)$$

The subscripts e and o indicate that the convolution kernels are even and odd. See Figure 2. Both filters are band-pass with center frequency approximately equal to Ω_x . In addition the filters are approximately Hilbert pairs, i.e. their frequency responses have nearly the same magnitude but are 90 degrees out of phase.

The outputs of these two filters can be computed by the linear CNN with second order cells described in [7]. The steady state values of the state variables $x_e(n, t)$ and $x_o(n, t)$ at each cell n correspond to the odd and even outputs of the spatial filtering stage. If the dynamics of the spatial filtering CNN are much faster than the temporal changes in the input, $x_e(n, t)$ and $x_o(n, t)$ are essentially equal to the input image convolved in space with the convolution kernels in (4).

Define $x(n, t) = x_e(n, t) + jx_o(n, t)$ where $j = \sqrt{-1}$. Due to the bandpass spatial filtering, the output at time 0 is approximately

$$x(n, 0) \approx a(n)e^{-j\Omega_x n}$$

where $a(n)$ varies slowly with n . If the image is translating with velocity $\bar{v}_n$ in the region of pixel n , then

$$x(n, t) \approx a(n - \bar{v}_n t)e^{j\Omega_x(n - \bar{v}_n t)} \approx (a(n)e^{j\Omega_x n})e^{-j\bar{v}_n \Omega_x t} \quad (5)$$

For fixed n and t increasing, $x(n, t)$ rotates around the origin of the complex plane with temporal frequency $\omega_t = -\bar{v}_n \times \Omega_x$. The direction of rotation (clockwise or counter-clockwise) depends on the direction of translation. The speed of rotation depends upon the speed of translation.

By feeding the output of the spatial filtering stage at pixel n into a bandpass temporal filter tuned to frequency Ω_{tn} , we obtain a spatio-temporal filter tuned to respond to velocities $v_n = -\Omega_{tn}/\Omega_x$. A convenient filter to use is a damped linear oscillator governed by the differential equation

$$\dot{y}(n, t) = (-\alpha + j\Omega_{tn})y(n, t) + \alpha x(n, t). \quad (6)$$

where α controls the bandwidth of the filter (i.e., the damping of the oscillator). Equivalently, defining $y(n, t) = y_e(n, t) + jy_o(n, t)$,

$$\begin{bmatrix} \dot{y}_e(n, t) \\ \dot{y}_o(n, t) \end{bmatrix} = \begin{bmatrix} -\alpha & -\Omega_{tn} \\ \Omega_{tn} & -\alpha \end{bmatrix} \begin{bmatrix} y_e(n, t) \\ y_o(n, t) \end{bmatrix} + \alpha \begin{bmatrix} x_e(n, t) \\ x_o(n, t) \end{bmatrix}.$$

Taking the temporal Fourier Transforms in (6),

$$\frac{Y(n, \omega_t)}{X(n, \omega_t)} = \frac{\alpha}{\alpha + j(\omega_t - \Omega_{tn})} = \frac{\alpha}{\sqrt{\alpha^2 + (\omega_t - \Omega_{tn})^2}} e^{j\phi} \text{ where } \phi = \text{atan}\left(\frac{\Omega_{tn} - \omega_t}{\alpha}\right).$$

If the approximation (5) is valid and the image velocity is equal to the tuned velocity $v_n = \bar{v}_n$, then the output $y(n, t)$ and input $x(n, t)$ are exactly in phase ($\phi = 0$). On the other hand, if $v_n \neq \bar{v}_n$, there is a phase shift between the output and the input given by

$$\phi = \text{atan}\left(\frac{\Omega_x}{\alpha}(\bar{v}_n - v_n)\right) \approx \frac{\Omega_x}{\alpha}(\bar{v}_n - v_n) \quad (7)$$

where the approximation is valid for $|\bar{v}_n - v_n|$ small. Thus, if the actual velocity is greater than the model velocity the output leads the input. If the actual velocity is less than the model velocity, the output lags the input. By measuring this phase shift, we can infer information about the image velocity. In fact, if we tune Ω_{tn} to zero the phase shift, we can estimate the image velocity by $\bar{v}_n \approx -\Omega_{tn}/\Omega_x$. Note that tunings of the filters at each pixel need not be identical. Thus, each filter could be tuned independently to estimate the image velocity in a local neighbourhood about each pixel. The size of the local neighbourhood is determined by the width of the convolution kernel in the spatial filtering stage. The smaller the width, the more local the estimate. On the other hand, the smaller the width, the wider the bandwidth of the spatial filter and the less valid the approximation in (5).

Another approximation for $|\bar{v}_n - v_n|$ which avoids the computation of the arctangent is

$$y_o(n, t)x_e(n, t) - x_o(n, t)y_e(n, t) = |x(n, t)| \cdot |y(n, t)| \cdot \sin \phi \approx |x(n, t)| \cdot |y(n, t)| \cdot \frac{\Omega_x}{\alpha}(\bar{v}_n - v_n) \quad (8)$$

where the approximation holds for $|\bar{v}_n - v_n|$ small. The ratio Ω_x/α is constant. The multiplicative factor $|x(n, t)| \cdot |y(n, t)|$ turns out to be a convenient confidence measure for the error signal. The larger $|x(n, t)|$, the more energy the input signal has at spatial frequency Ω_0 and the more reliable the velocity information. The larger $|y(n, t)|$, the closer the model velocity is to the actual velocity and the better the approximation in (8).

4. Estimation of Time to Contact

Using an adaptive filter at each pixel, the image velocity can be independently estimated at each pixel. However, this estimate may be unreliable at pixels where the confidence measure $|x(n, t)| \cdot |y(n, t)|$ is small. This section describes how to combine information over the entire image to estimate the time to contact under the assumptions in Section 2.

Suppose we have an N pixel image. If the assumptions in Section 2 are valid, image velocity is a linear function of position and the time to contact is inversely proportional to the slope. If we only have noisy estimate of the

Figure 3: Circuit for least squares fitting. At steady state, the voltages v_0 and v_{N+1} settle so that the voltages v_1 to v_N are the best linear fit to the data $\bar{v}_1$ to $\bar{v}_N$ in the least squares sense. The triangular blocks represent op amps. The trapezoidal blocks represent linear differential transconductance amplifiers.

velocity at each pixel, a robust estimate of the slope of the velocity variation is the slope of a linear least squares fit to the data.

A circuit which performs this linear least squares fitting is shown in Figure 3. The line of resistors at the top of the figure generate the voltages v_1 to v_N by linearly interpolating between v_0 and v_{N+1} ,

$$v_n = v_0 + (v_{N+1} - v_0) \frac{n}{N+1}.$$

At steady state, these voltages represent the linear least squares fit to the data voltages $\bar{v}_n$ at the input to the transconductance amplifiers. To see this, define the squared error between the linear fit and the data by

$$E = \frac{1}{2} \sum_{i=1}^N (v_i - \bar{v}_i)^2.$$

To reduce the error we can update v_0 and v_{N+1} in the direction opposite to the derivatives

$$\begin{aligned} \frac{dE}{dv_0} &= \sum_{n=1}^N (v_n - \bar{v}_n) \frac{dv_n}{dv_0} = \sum_{n=1}^N (v_n - \bar{v}_n) \frac{n+1-i}{N} \\ \frac{dE}{dv_{N+1}} &= \sum_{n=1}^N (v_n - \bar{v}_n) \frac{dv_n}{dv_{N+1}} = \sum_{n=1}^N (v_n - \bar{v}_n) \frac{n}{N+1} \end{aligned}$$

Using superposition it can be shown that the currents leaving the two ends of the lower resistive line are proportional to these derivatives. By integrating these currents using op-amp based integrators, the voltages v_0 and v_{N+1} settle to values such that the error E is minimized. If the voltages in the circuit represent the velocities in the image, the time to contact can be estimated by

$$\frac{1}{t_c} = \frac{v_{N+1} - v_0}{N+1}.$$

It turns out that we do not need to estimate the velocity at each pixel explicitly. Instead we can use the voltages v_n to control the tuning frequencies of the temporal filters at each pixel as described in Section 3 and can replace the currents generated by the differential transconductance amplifiers by the phase difference signal in (8). If the differences $|v_n - \bar{v}_n|$ are small, then the cost function being minimized at each time t is

$$E(t) = \frac{1}{2} \sum_{n=1}^N |x(n, t)| \cdot |y(n, t)| \cdot (v_n - \bar{v}_n)^2$$

Since $E(t)$ changes over time, stability is not guaranteed by Lyapunov-type arguments. However, this cost function has the advantage that locations at which there is more spatial frequency information from which to estimate the velocity are weighted more than those where there is little spatial frequency information. The disadvantage of this update is that the time scale of the dynamics is dependent upon the image contrast. The greater the image contrast, the faster the dynamics.

Figure 4: (a) The sequence of one dimensional input images for a camera approaching a textured planar surface with constant time to contact. (b) The time evolution of velocity estimates v_0 and v_{N+1} . The actual values are -0.2 and 0.2 pixels per frame. The initial estimates at time 0 are 0.2 and -0.2 pixels per frame.

5. Computer Simulation Results

In this section, we demonstrate by computer simulation that for images where the velocity variation across the image is linear, the algorithm described above can estimate the velocity variation. Since the computations must be done in discrete time, we replace the continuous time filters by discrete time filters governed by difference equations of the form

$$y(n, k+1) = (ae^{-jv_n \Omega_x})y(n, k) + (1-a)u(n, k+1)$$

and the endpoint velocity estimates $v_0(k)$ and $v_{N+1}(k)$ are updated according to

$$v_0(k+1) = v_0(k) + \mu \cdot \Delta v_0(k)$$

$$v_{N+1}(k+1) = v_{N+1}(k) + \mu \cdot \Delta v_{N+1}(k)$$

where

$$\Delta v_0(k) = \sum_{n=1}^N (y_o(n, k)x_e(n, k) - x_o(n, k)y_e(n, k)) \frac{n+1-i}{N}$$

$$\Delta v_{N+1}(k) = \sum_{n=1}^N (y_o(n, k)x_e(n, k) - x_o(n, k)y_e(n, k)) \frac{n}{N+1}$$

Otherwise, the algorithm is identical to the continuous time algorithm described above. The parameters used in the simulation are shown in Table 1. Note that due to the dependency of the size of $\Delta v_0(k)$ and $\Delta v_{N+1}(k)$ on the image contrast, for images with very large contrast, the velocity update may be unstable.

The results shown in Figure 4 assume that the time to contact in the image is constant. This implies that the velocity of the camera as it approaches the planar object is decreasing. The synthetic sequence of one dimensional input images is shown in Figure 4(a) as a two dimensional image where the time axis runs vertically down the image. The image velocity is constant in time but varies linearly in space from -0.2 pixels/frame on the left edge to 0.2 pixels/frame on the right edge. Although the initial velocity estimates of the image velocities at the edge pixels of the array are actually opposite in sign to the actual velocities, the estimates converge smoothly to the actual values as seen in Figure 4(b).

The results shown in Figure 5 assume that the camera is approaching an object with constant velocity. Thus, the time to contact decreases over time while and the slope of the velocity increases. The synthetic sequence of one dimensional input images is shown in Figure 5(a) as a two dimensional image where the time axis runs vertically down the image. The image velocity increases over time to minimum and maximum values of -0.2 pixels/

Figure 5: (a) The sequence of one dimensional input images for a camera approaching a textured planar surface with constant velocity. (b) The time evolution of velocity estimates v_0 and v_{N+1} are shown as dotted lines. The actual values are shown as solid lines. The initial estimates at time 0 are 0.01 and -0.01 pixels per frame.

frame on the left edge and 0.2 pixels/frame on the right edge. Despite an initial estimation error, the estimates converge to and track the actual values as seen in Figure 5(b).

Stage	Parameters
Spatial Filtering	$\lambda = 0.15$, $\Omega_x = 2 \text{atan} 0.5$
Temporal Filtering	$a = 0.8$

Table 1: Simulation Parameters

6. Conclusion

This paper has described the application of second order CNN arrays for spatial and spatio-temporal image filtering to estimation of the time to contact in a one dimensional image taken by a camera approaching a planar object which is parallel to time image plane. Experimental results on synthetic images indicate that the approach gives accurate estimates of the time to contact. Current and future research includes more extensive testing of the algorithm on natural images and the hardware implementation of the algorithm in analog VLSI.

References

- [1] D. T. Lawton, "Processing translational motion sequences," *Computer Vision, Graphics and Image Processing*, vol. 22, pp. 116-141, 1983.
- [2] M. Tistarelli and G. Sandini, "Robot navigation using an anthropomorphic visual sensor," Proceedings of the IEEE International Conference on Robotics and Automation, pp. 374-383, 1990.
- [3] L. O. Chua and L. Yang, "Cellular Neural Networks: Theory," *IEEE Transactions on Circuits and Systems*, vol. 35, pp. 1257-1272, Oct. 1988.
- [4] L. O. Chua and L. Yang, "Cellular Neural Networks: Applications," *IEEE Transactions on Circuits and Systems*, vol. 35, pp. 1273-1290, Oct. 1988.
- [5] L. O. Chua and T. Roska, "The CNN paradigm," *IEEE Transactions on Circuits and Systems - I: Fundamental Theory and Applications*, vol. 40, pp. 147-156, Mar. 1993.
- [6] B. K. P. Horn, *Robot Vision*, The MIT Press, Cambridge, MA, 1986.
- [7] B. E. Shi, "Gabor-type image filtering with cellular neural networks," *Proceedings of the 1996 IEEE International Symposium on Circuits and Systems*, May 13-15, 1996, to appear.
- [8] B. E. Shi, "Spatio-temporal image filtering with cellular neural networks," *Proceedings of the 1996 International Conference on Neural Networks*, June 2-6, 1996, to appear.