Canny Edge Detector

Common Names: Canny edge detector

Brief Description

The Canny operator was designed to be an optimal edge detector (according to particular criteria --- there are other detectors around that also claim to be optimal with respect to slightly different criteria). It takes as input a gray scale image, and produces as output an image showing the positions of tracked intensity discontinuities.

How It Works

The Canny operator works in a multi-stage process. First of all the image is smoothed by Gaussian convolution. Then a simple 2-D first derivative operator (somewhat like the Roberts Cross) is applied to the smoothed image to highlight regions of the image with high first spatial derivatives. Edges give rise to ridges in the gradient magnitude image. The algorithm then tracks along the top of these ridges and sets to zero all pixels that are not actually on the ridge top so as to give a thin line in the output, a process known as *non-maximal suppression*. The tracking process exhibits hysteresis controlled by two thresholds: *T1* and *T2*, with *T1* > *T2*. Tracking can only begin at a point on a ridge higher than *T1*. Tracking then continues in both directions out from that point until the height of the ridge falls below *T2*. This hysteresis helps to ensure that noisy edges are not broken up into multiple edge fragments.

Guidelines for Use

The effect of the Canny operator is determined by three parameters --- the width of the Gaussian kernel used in the smoothing phase, and the upper and lower thresholds used by the tracker. Increasing the width of the Gaussian kernel reduces the detector's sensitivity to noise, at the expense of losing some of the finer detail in the image. The localization error in the detected edges also increases slightly as the Gaussian width is increased.

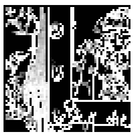
Usually, the upper tracking threshold can be set quite high, and the lower threshold quite low for good results. Setting the lower threshold too high will cause noisy edges to break up. Setting the upper threshold too low increases the number of spurious and undesirable edge fragments appearing in the output.

One problem with the basic Canny operator is to do with Y-junctions *i.e.* places where three ridges meet in the gradient magnitude image. Such junctions can occur where an edge is partially occluded by another object. The tracker will treat two of the ridges as a single line segment, and the third one as a line that approaches, but doesn't quite connect to, that line segment.

We use the image

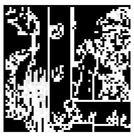
to demonstrate the effect of the Canny operator on a natural scene.

Using a Gaussian kernel with standard deviation 1.0 and upper and lower thresholds of 255 and 1, respectively, we obtain



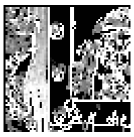
Most of the major edges are detected and lots of details have been picked out well --- note that this may be too much detail for subsequent processing. The `Y-Junction effect' mentioned above can be seen at the bottom left corner of the mirror.

The image



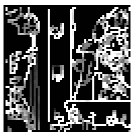
is obtained using the same kernel size and upper threshold, but with the lower threshold increased to 220. The edges have become more broken up than in the previous image, which is likely to be bad for subsequent processing. Also, the vertical edges on the wall have not been detected, along their full length.

The image



is obtained by lowering the upper threshold to 128. The lower threshold is kept at 1 and the Gaussian standard deviation remains at 1.0. Many more faint edges are detected along with some short `noisy' fragments. Notice that the detail in the clown's hair is now picked out.

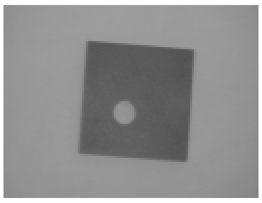
The image



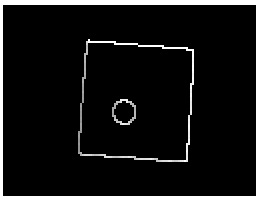
is obtained with the same thresholds as the previous image, but the Gaussian used has a standard deviation of 2.0. Much of the detail on the wall is no longer detected, but most of the strong edges remain. The edges also tend to be smoother and less noisy.

Edges in artificial scenes are often sharper and less complex than those in natural scenes, and this generally improves the performance of any edge detector.

The image



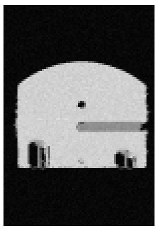
shows such an artificial scene, and



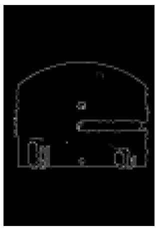
is the output from the Canny operator.

The Gaussian smoothing in the Canny edge detector fulfills two purposes: first it can be used to control the amount of detail that appears in the edge image and second, it can be used to suppress noise.

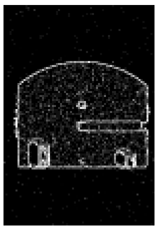
To demonstrate how the Canny operator performs on noisy images we use



which contains Gaussian noise with a standard deviation of *15*. Neither the Roberts Cross nor the Sobel operator are able to detect the edges of the object while removing all the noise in the image. Applying the Canny operator using a standard deviation of *1.0* yields

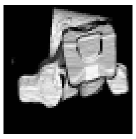


All the edges have been detected and almost all of the noise has been removed. For comparison,

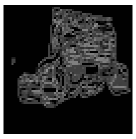


is the result of applying the Sobel operator and thresholding the output at a value of *150*.

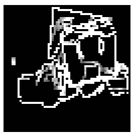
We use



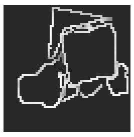
to demonstrate how to control the details contained in the resulting edge image. The image



is the result of applying the Canny edge detector using a standard deviation of *1.0* and an upper and lower threshold of *255* and *1*, respectively. This image contains many details; however, for an automated recognition task we might be interested to obtain only lines that correspond to the boundaries of the objects. If we increase the standard deviation for the Gaussian smoothing to *1.8*, the Canny operator yields

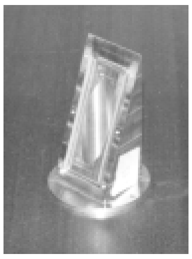


Now, the edges corresponding to the uneveness of the surface have disappeared from the image, but some edges corresponding to changes in the surface orientation remain. Although these edges are `weaker' than the boundaries of the objects, the resulting pixel values are the same, due to the saturation of the image. Hence, if we scale down the image before the edge detection, we can use the upper threshold of the edge tracker to remove the weaker edges. The image

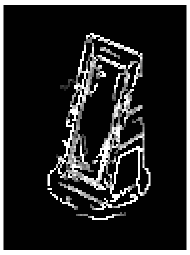


is the result of first scaling the image with *0.25* and then applying the Canny operator using a standard deviation of *1.8* and an upper and lower threshold of *200* and *1*, respectively. The image shows the desired result that all the boundaries of the objects have been detected whereas all other edges have been removed.

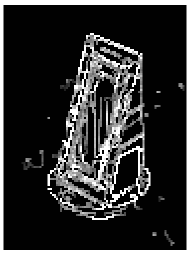
Although the Canny edge detector allows us the find the intensity discontinuities in an image, it is not guaranteed that these discontinuities correspond to actual edges of the object. This is illustrated using



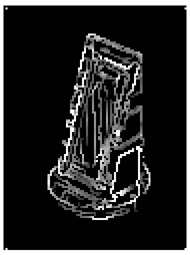
We obtain



by using a standard deviation of *1.0* and an upper and lower threshold of *255* and *1*, respectively. In this case, some edges of the object do not appear in the image and many edges in the image originate only from reflections on the object. It is a demanding task for an automated system to interpret this image. We try to improve the edge image by decreasing the upper threshold to *150*, as can be seen in



We now obtain most of the edges of the object, but we also increase the amount of noise. The result of further decreasing the upper threshold to *100* and increasing the standard deviation to *2* is shown in



Common Variants

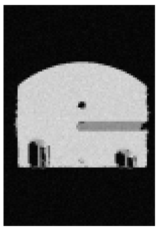
The problem with Y-junctions mentioned above can be solved by including a model of such junctions in the ridge tracker. This will ensure that no spurious gaps are generated at these junctions.

Interactive Experimentation

You can interactively experiment with this operator by clicking here.

Exercises

1. Adjust the parameters of the Canny operator so that you can detect the edges of

   while removing *all* of the noise.

2. What effect does increasing the Gaussian kernel size have on the magnitudes of the gradient maxima at edges? What change does this imply has to be made to the tracker thresholds when the kernel size is increased?

3. It is sometimes easier to evaluate edge detector performance after thresholding the edge detector output at some low gray scale value (*e.g.* 1) so that all detected edges are marked by bright white pixels. Try this out on the third and fourth example images of the clown mentioned above. Comment on the differences between the two images.

4. How does the Canny operator compare with the Roberts Cross and Sobel edge detectors in terms of speed? What do you think is the slowest stage of the process?

5. How does the Canny operator compare in terms of noise rejection and edge detection with other operators such as the Roberts Cross and Sobel operators?

6. How does the Canny operator compare with other edge detectors on simple artificial 2-D scenes? And on more complicated natural scenes?

7. Under what situations might you choose to use the Canny operator rather than the Roberts Cross or Sobel operators? In what situations would you definitely not choose it?

References

R. Boyle and R. Thomas *Computer Vision: A First Course*, Blackwell Scientific Publications, 1988, p 52.

J. Canny *A Computational Approach to Edge Detection*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 8, No. 6, Nov. 1986.

E. Davies *Machine Vision: Theory, Algorithms and Practicalities*, Academic Press, 1990, Chap. 5.

R. Gonzalez and R. Woods *Digital Image Processing*, Addison-Wesley Publishing Company, 1992, Chap. 4.

Local Information

Specific information about this operator may be found here.

More general advice about the local HIPR installation is available in the *Local Information* introductory section.