Computer Vision 558
Corner Detection Overview and Comparison

Alexandar Alexandrov
ID 9823753

May 3, 2002

Contents

1 Introduction **2**
- 1.1 How it started . . . 2
- 1.2 Playing with ideas . . . 2
- 1.3 Previous work . . . 3

2 Corner Detectors **4**
- 2.1 Requirements . . . 4
- 2.2 The Plessey Feature Point Detector . . . 4
- 2.3 The SUSAN detector . . . 5
- 2.4 The Curvature Scale Space (CSS) Corner Detector . . . 6
- 2.5 IPAN99 . . . 7

3 Discussion **8**

4 Conclusion **9**

5 Visual Image Results **11**

1 Introduction

1.1 How it started

My original idea was to implement the beginning stages of a paper on obtaining 3D models from a sequence of images [1]. This has been an ongoing research topic in Computer Vision. There is demand for ability to reconstruct the surrounding world just by looking at it just the way humans do it. Applications can range from robot vision to virtual reality models and communication. Originally (some systems still use it) 3D reconstruction was done with expensive hardware like laser range finders for example, but having a cheaper alternative makes it more accessible to the average user. More and more people find useful applications of it, encouraging the further research of those new models. The basic idea behind 3D Model reconstruction from a video (or a sequence of un-calibrated images) could be defined in several steps: first we need to relate the images in the whole sequence, then extract information on pixel correspondence in order to apply methods from epipolar geometry. Projectively, the epipolar geometry is all there is to know about a stereo rig [2]. It establishes the correspondence, and allows 3D reconstruction of the scene to be carried out up to an overall 3D projective deformation (which is all that can be done with any number of completely un-calibrated cameras, without further constraints). The overall 3D projective deformation can be reduced through self-calibration procedures to produce a metric 3D model of the object or scene at hand. An accurate depth map can be calculated from this metric Model later on.

I found it interesting to figure out how exactly the images are related and the problems of correspondence are solved prior to using epipolar geometry. It is not feasible to compare every pixel of one image with every pixel of the next one. The complexity is too high and as I have found out implementing a brute force pattern synthesis algorithm [3]. In real life image sequences there are a lot of points that are better suited for automated matching than others. The neighborhood of some points contain a lot of intensity variations and are therefore easy to differentiate from others. An interest point detector is used to select a certain number of such suited points. The correspondence between such points of interest has to be done using a matching procedure.

1.2 Playing with ideas

At this point finding interesting points in an image became literally more interesting than actually relating them later on. As it turns out a decision on local features is still not a closed issue. Many algorithms have their advantages and disadvantages. One of the proposed ways to selecting points of interest is corner detection. The idea is that if there is a corner in the scene. then it is the end of an edge. And an end of an edge usually defines a boundary between two different objects or between parts of the same object. For the above 3D model reconstruction algorithm, for example,if we are able to select points on the corners of objects we have a greater chance of matching the same corner

taken from the next image. One was suggested in [1] the Harris corner detector [4] which they used for selecting feature points in their implementation.

Naturally [4] is not the only one available but are they all equally good? Can their results be improved using different parameters? Is it possible to detect all corners in a real life image? Which ones do we use in what situation? How different are they from one another? What are their basic principles? I will try to find answers to these questions based on previous works on the subject. Of course and attempt to obtain working versions of all of them was made and where possible used, but after attempting to install several different ones on our lab machines with little success I got a little bit discouraged and decided to reuse other people's test results.

1.3 Previous work

Given a digital image, the earliest geometry-based corner detectors first segment the sharp, extract its boundary as chain code, and then search for significant turnings at the boundary. This kind of corner detectors suffers the high algorithm complexity, as multiple steps are needed. What's more, any errors in the segmentation step will lead to great astray in the corner results. Afterward, many geometry-based corner detectors, which directly operate on the gray level image, were introduced. Kitchen found that the change of gradient direction multiplied by the local gradient could isolate the corners well. Wang and Brady observed that the total curvature of the gray level image is proportional to edge normal, and inversely proportional to the edge strength. Moravec proposed an interest point detector which functions by considering a local window in an image and determining the average changes of intensity which results from shifting the window by a small amount in four directions. Harris and Stephens[4] modified the Moravec method to the Plessey corner detector by estimating the autocorrelation from the first-order derivatives. Most of them are sensitive to noise as they depend on the image derivatives. Smith and Brady introduced a straightforward method named SUSAN that does not depend on image derivatives [6]. Given a digital image, the USAN (Univalue Segment Assimilating Nucleus) area will reach a minimum when the nucleus lies on a corner point. SUSAN is not sensitive to noise and it is very fast as it only uses very simple operations.

Previous research and comparison between corner detectors has also been done in [8] where the performance of four different algorithms was tested on affine and scale transformation of the same image. [5] gives a brief description on probably all existing corner detection existing up to 1997.

2 Corner Detectors

2.1 Requirements

Corner detection should satisfy a number of important criteria:

- All the true corners should be detected.
- No false corners should be detected.
- Corner points should be well localized.
- Corner detector should be robust with respect to noise.
- Corner detector should be efficient.

2.2 The Plessey Feature Point Detector

In [4] Harris and Stephens described what has become known as the Plessey feature point detector. We can see the outline of how it works if we consider the following matrix:

$$\mathbf{M} = \begin{bmatrix} \left(\frac{\partial I}{\partial x}\right)^2 & \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) \\ \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) & \left(\frac{\partial I}{\partial y}\right)^2 \end{bmatrix} \tag{1}$$

where $I(x, y)$ is the grey level intensity. If at a certain point the two eigenvalues of the matrix $\mathbf{M}$ are large, then a small motion in any direction will cause an important change of grey level. This indicates that the point is a corner. The corner response function is given by:

$$R = \det \mathbf{M} - k(\text{trace}\mathbf{M})^2 \tag{2}$$

where k is a parameter set to 0.04 (a suggestion of Harris). Corners are defined as local maxima of the corner-ness function. Sub-pixel precision is achieved through a quadratic approximation of the neighborhood of the local maxima. To avoid corners due to image noise, it can be interesting to smooth the images with a Gaussian filter. This should however not be done on the input images, but on images containing the squared image derivatives (i.e. $\left(\frac{\partial I}{\partial x}\right)^2$, $\left(\frac{\partial I}{\partial y}\right)^2$, $\left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right)$).

In practice often far too much corners are extracted. In this case it is often interesting to first restrict the numbers of corners before trying to match them. One possibility consists of only selecting the corners with a value R above a certain threshold. This threshold can be tuned to yield the desired number of features. Since for some scenes most of the strongest corners are located in the same area, it can be interesting to refine this scheme further to ensure that in every part of the image a sufficient number of corners are found.

2.3 The SUSAN detector

SUSAN (Smallest Univalue Segment Assimilating Nucleus) presents us with an entirely different approach to low level image processing compared to all pre-existing algorithms. It provides corner detection as well as edge detection and is more resistant to image noise although no noise reduction (filtering) is needed.

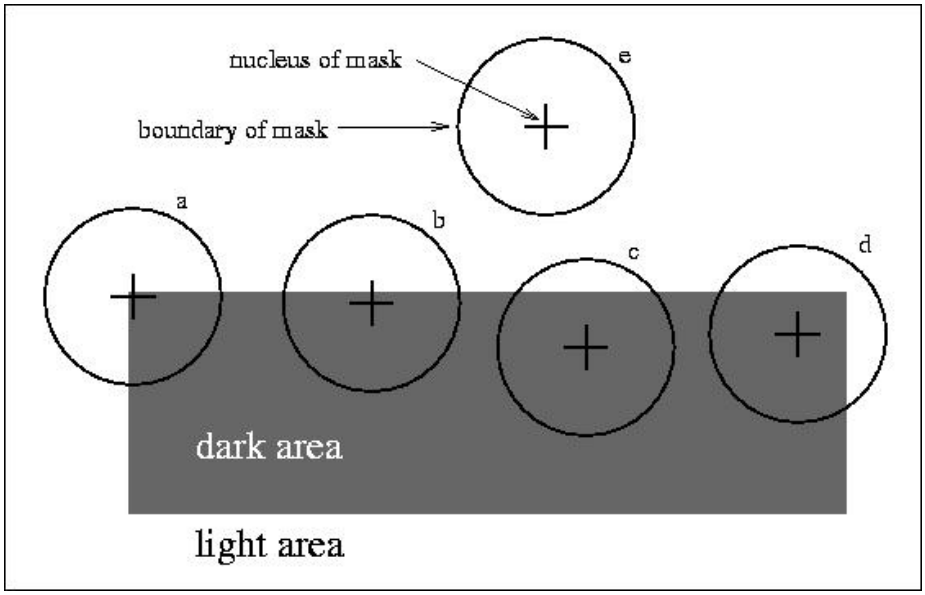

Figure 1: Four circular masks at different places on a simple image.

The concept of each image point having associated with it a local area of similar brightness is the basis for the SUSAN principle. If the brightness of each pixel within a mask is compared with the brightness of that mask's nucleus then an area of the mask can be defined which has the same (or similar) brightness as the nucleus. This area of the mask shall be known as the"USAN", an acronym standing for "Univalue Segment Assimilating Nucleus". In Figure 2 each mask from Figure 1 is depicted with its USAN shown in white.

Computing USAN for every pixel in the digital image provides us with a way to determine the edges inside it. The value of USAN gets smaller on both sides of an edge and becomes even smaller on each side of a corner. Hence we are looking for the Smallest USAN (or SUSAN for short). The local minima of the USAN map represents corners in the image.

The reason that this method stays resistant to noise is the lack of computing spatial derivatives of the image intensity.

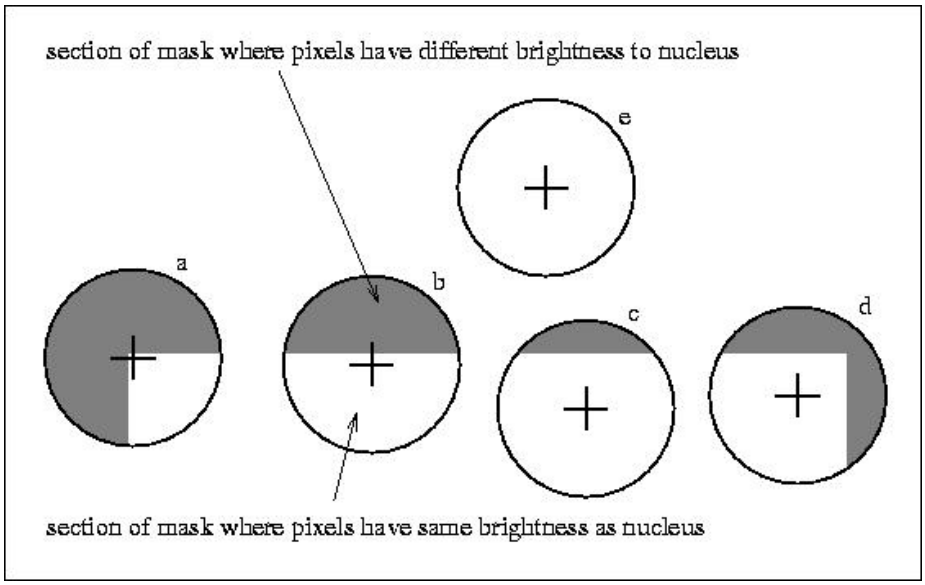

Figure 2: Four circular masks with similarity colouring; USANs are shown as the white parts of the masks.

2.4 The Curvature Scale Space (CSS) Corner Detector

The curvature scale space technique is suitable for recovering invariant geometric features (curvature zero-crossing points and/or extrema) of a planar curve at multiple scales. The CSS corner detector works as follows [7]:

- Extract the edge contours from the input image using any good edge detector such as Canny.
- Fill small gaps in edge contours. When the gap forms a T-junction, mark it as a T-corner.
- Compute curvature on the edge contours at a high scale.
- The corner points are defined as the maxima of absolute curvature that are above a threshold value.
- Track the corners through multiple lower scales to improve localization.
- Compare T-corners to the corners found using the CSS procedure and remove very close corners.

Experimental results show this algorithm spends most of its time (80%) detecting the edges in the image. Faster edge detectors may be used. The local maxima of absolute curvature are the possible candidates for corner points. A

local maximum is either a corner, the top value of a rounded corner or a peak due to noise. The latter two should not be detected as corners. The curvature of a real corner point has a higher value than that of a rounded corner or noise. The corner points are also compared to the two neighboring local minima. The curvature of a corner should be twice that of one of the neighboring local minima. This is because when the shape of the contour is very round, contour curvature can be above the threshold.

It is interesting to see that in this method two types of corners are detected: T shaped and and corners on the curve. we have to avoid assigning two corners to pixels which are close together. Figure 3 illustrates when this type of problem arises. This explains the reason fot the last step in the CSS algorithm above.

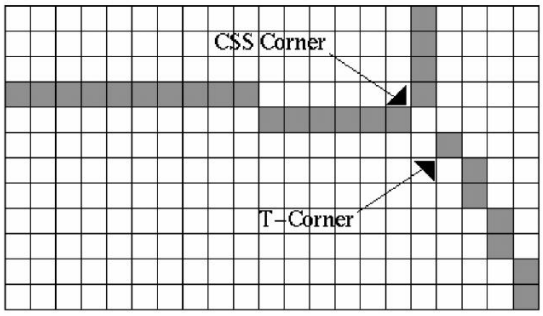

Figure 3: Case where one corner is marked twice

2.5 IPAN99

The acronym for this method stands for "Image and Pattern Analysis group" and was developed in 1999 at The Hungary Academy of Science. It is fast and efficient algorithm for detection of high curvature points. It works in two passes and defines a corner in a simple and intuitively appealing way, as a location where a triangle of specified size and opening angle can be inscribed in a curve. A curve is represented by a sequence of points which a densely sampled along the curve but are not in a uniform distance from one another. The curve has to be generated previously using an edge detector. It is not required to be a closed curve. In the first pass the sequence of points is scanned and candidate corner points are selected. In each curve point p the detector tries to inscribe in the curve a variable triangle (p^- ,p ,p^+). The triangle varies between a minimum and a maximum square distance on the curve from p^- to p, from p to p^+ and the angle $\alpha \leq \alpha_{max}$ between the two lines a and b in Figure 4 a). Because the points are kept with their Cartesian coordinates we can easily compute α. Triangles are selected starting from point p outward and stop on the conditions mentioned above. In that way a number of admissible triangles are defined. At a neighborhood of points only one of these admissible triangles is selected - the

one which has the smallest value for α. A value *sharpness* is assigned to p.

In the second pass the selection is refined and points that give the strongest response are marked as corners in the curve. this is done by selecting only points which have *sharpness* greater than that of their neighbors.

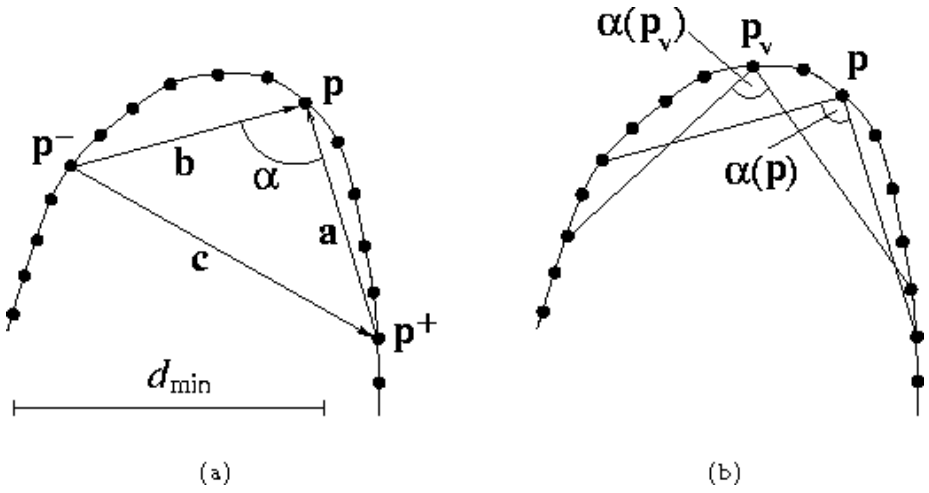

Figure 4: Detecting high curvature points. (a) Determining if p is a candidate point. (b) Testing p for sharpness

3 Discussion

Although the notion of corner seems to be intuitively clear, no generally accepted mathematical definition exists, at least for digital curves. In a sense, different approaches give different, but conceptually related computational definitions to a visual phenomenon. The four different methods discussed in the previous section could be classified as representatives of different mainstream corner detectors. They differ in their approach to the problem. Plessey uses calculus and image intensities to detect sudden changes in the first order image derivatives. The eigenvectors provide a match for corners at different angles. Compared to all the others Plessey performs very poorly. This is partially due to the fact that it uses no information from the rest of the image namely global edges and operates only at the local neighborhood of the pixel being examined. Although it uses a Gaussian filter it is also not robust to noise. Localization accuracy is also poor, particularly at certain junction types. This occurs for two reasons. The first is that the method can only work accurately at most junctions if the masks used to find the derivatives and to perform smoothing have zero size. This can be easily understood in the case of "T" junctions, where the two "smaller" regions at the junction have similar brightness, and the "larger"

region has a very different brightness from the other two. The other reason for localization error is that the actual measure of interest depends to a large extent on the relative values of the two curvatures, and is greatest when they are equal. In the case described above, the two curvatures will be equal at some point along the weak edge, away from the strong edge, and therefore away from the correct position of the junction [5].

It might seem that the SUSAN algorithm works taking its information only by looking at the local neighborhood (the circular Mask) it actually implements an precise edge detector. The intuitive notion of a corner is the meting of two or more edges in one point.The edges separate the image into regions. Thus at junctions involving more than two regions, for example, "T" junctions, more than one region may contribute to the detection of a "corner", and all junctions will be correctly processed, no matter how complicated they are. Also, no assumptions about the internal structures of the regions around a "corner" are made; it is common to assume that the regions are constant in value, but in the case of the SUSAN detector, sloping regions are allowed. the results from the SUSAN detector clearly show that clustering of corners is allowed on a small region of the image. this happens because unlike all other detectors that i examined, SUSAN does not average between the points of interest found. The only reason it might need to is in case of image noise.

CSS is a more modern model using curve evolution. The corners are defined as the local maxima of the absolute value of the curvature. At a very fine scale there exists many such maxima due to noise on the digital contour. As the scale is increased the noise is smoothed away and only the maxima corresponding to the real corners remain. The CSS detector gives the best results out of the ones tested. Calibration and changing of parameters do not have much effect on the quality and number of corners detected. All the tested reviewed Detector get a number of parameters as thresholds for closer and better matching. For IPAN these parameters would be the easiest to visualize the results reason one is that they are closer to the intuitive idea of corners. If we think of a triangle then we specify the maximum angle of one of its corners. exceeding this value makes the corner appear as a straight line.

The Most intuitive Idea for Edge detection for me is the IPAN algorithm although it does not give as good results as CSS. It is designed originally for detecting single objects and representing the whole image as one curve. Which of course cannot be applied to all situations. IPAN and CSS both use curve and curvature but while CSS evolves the curve IPAN keeps it static, only inscribing admissible triangles.

4 Conclusion

While reviewing the models and looking for others some ideas came in mind. No precise Corner detector exists, otherwise all will be using it and the issue will be closed. All detect corners to some extend, not all are invariant to image noise although there are methods, applying filters to the image if we expect

it to be noisy. For correspondence it is important that if a detector finds n interesting points in one image, hopefully n interesting points will be found in the next. This means that a detector needs not to be so much accurate but more consistent. because it will find the same points interesting in both images from the sequence provided they are not all that different. For other areas like recognition though a falsely detected corner will lead to a false model of the observed object and precision is necessary.

Since we have a number of good and not so good corner detectors (I read about a lot more than these four) and they all find different points and "claim" them as corners we can combine several of them. If we need precision we can run 10 of the *better* ones on the required data, keeping the results separate and then combine them averaging out the points that get less hits. The more diverse the detectors used the higher accuracy achieved with minimum to no adjustments in the parameters. I did not come accross any other paper suggesting such a method although at that stage it looks obvious to me that this will at least be an interesting experiment.

5 Visual Image Results

Images were taken from[11] and [10]

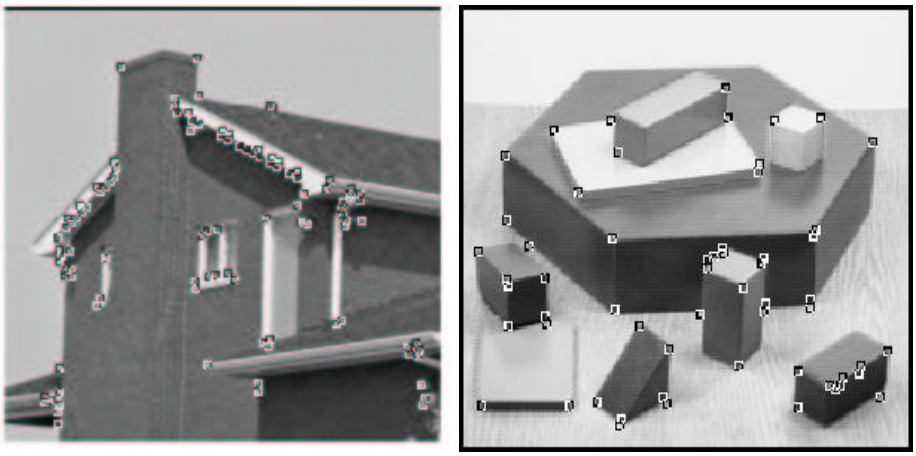

Figure 5: Plessey.

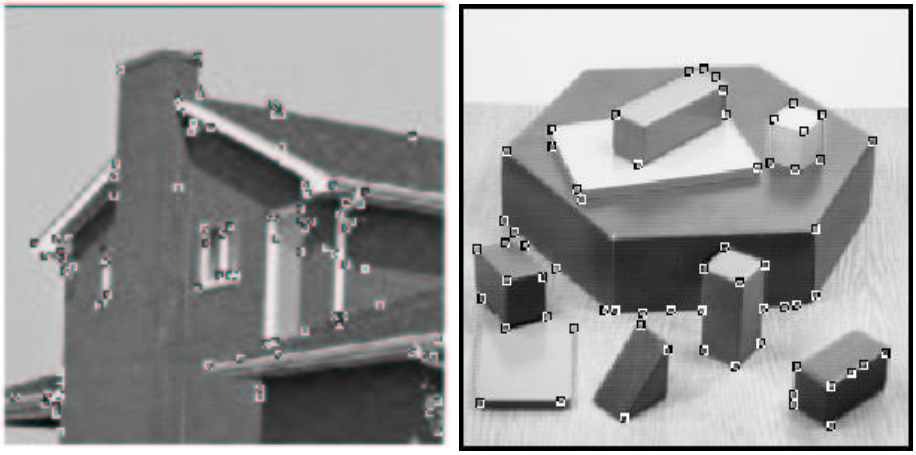

Figure 6: SUSAN.

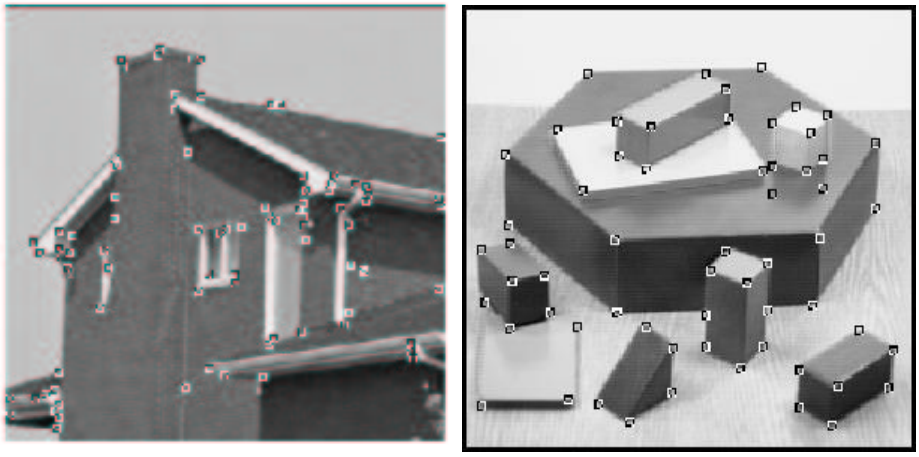

Figure 7: CSS.

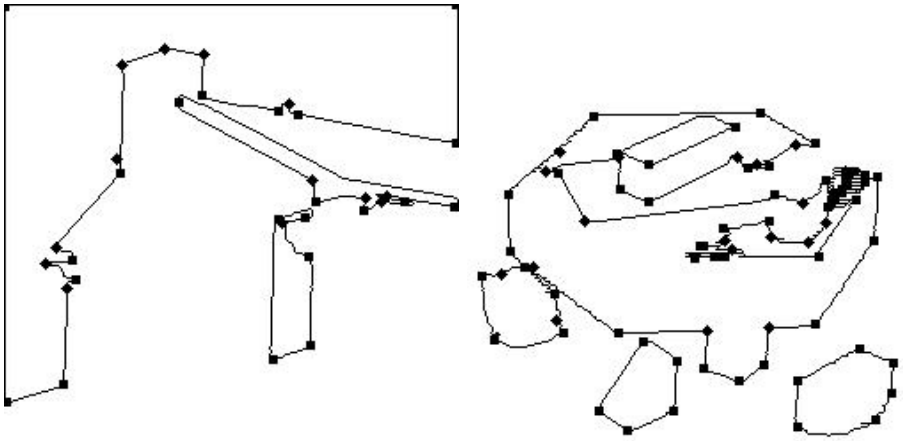

Figure 8: IPAN99

References

[1] Mark Polleyfeys, Reinard Koch, Maarten Vergauwen, and Luc Van Gool *Metric 3d Surface Reconstruction from Uncalibrated Image Sequences*

[2] Roger Mohr and Bill Triggs. *Tutorial on projective geometry given at International Symposium of Photogrammetry and Remote Sensing, Vienna 1996*

[3] Alexei A. Efros and Thomas K. Leung *Texture Synthesis by Non-parametric Sampling*

[4] C. Harris and M. Stephens, *A combined corner and edge detector", Fourth Alvey Vision Conference, pp.147-151, 1988.*

[5] Stephen M. Smith, *Reviews of Optic Flow, Motion Segmentation.*

[6] S.M. Smith and J.M. Brady. *SUSAN - a new approach to low level image processing*

[7] Mokhtarian, F. and R. Suomela, *Curvature Scale Space Based Image Corner Detection, Proc. European Signal Processing Conference, pp. 2549-2552, Island of Rhodes, Greece, 1998.*

[8] F. Mohanna and F Mokhatarian, *Performance Evaluation of Corner Detection Algorithms Using Similarity and Affine Transforms*

[9] Dmitry Chetverikov and Zsolt Szab *Detection of High Curvature Points in Planar Curves*

[10] http://visual.ipan.sztaki.hu/corner/

[11] http://www.ee.surrey.ac.uk/Research/VSSP/demos/corners/results3.html