WORKING NOTE AC122

Alan Creak
1999 April 8

THE DISTORTION NETWORK

This note is for the most part a compilation of previous miscellaneous notes on the subject of the distortion network, with bits added so that it makes a sort of sense. The original notes were written for research students (Duncan Sharp in 1995 and Mark Scaletti in 1993). I have tidied them up, sorted out differences in nomenclature, and weeded out inconsistencies when I've seen them. In the process, a few mildly new ideas have come along, so they're here too along with the old stuff.

THE IDEA.

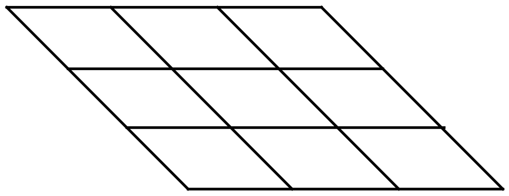
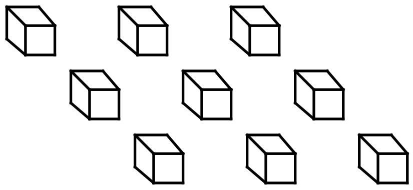
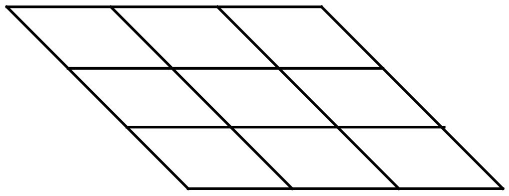
People recognise things by comparing some representation of what we see with some sort of set of stored templates. The matching is such that some degree of distortion is tolerated, so that some sorts of distortion are handled quite automatically. We cope with perspective, different sizes, occlusion, and naturally variable objects as a matter of course. To reproduce this exceedingly valuable ability in a mechanical system, we would like a matching technique which was inherently similarly tolerant of distortion. The distortion network is suggested as a means of implementing such a mechanical matching system using a parallel connectionist structure.

The aim of the network is to find a mapping of a standard image to a specimen image which causes the two images to correspond well with minimum distortion. The behaviour of the network is determined by the interaction of the distortion with the good matching between standard and specimen images.

THE MACHINERY.

The essential **architecture** is rather simple. (Note that the *essential* architecture doesn't include displays, or anything else which is nice but not directly related to the business of the distortion network itself.)

A simple distortion network can conveniently be thought of as composed of three layers; the diagram below illustrates the architecture and defines some nomenclature.

Name of layer		*Contents of layer*	*Nature of layer*
Pattern layer		Standard image	A plane of passive pixels
Matching layer		Redirection information	A plane of active processors
Retina		Specimen image	A plane of passive pixels

The pattern layer is the network's memory, recording the standard image to be matched as a square array of pixels. The pixels are binary, at least for the moment, for reasons which are discussed below. The retina, which is the same size as the pattern layer and identical in construction, records the specimen image for inspection. The matching layer lies between the other two and is the only active component; this is the clever bit. Its function is to determine whether or not there is a version of the standard image in the specimen, and, if so, to match the corresponding parts of the two images.

So far as the network is concerned, the identification of one of the images as the standard (in effect, the memory of how an image should be) and the other as a specimen is arbitrary; so far as I can see, it doesn't matter which way round we run the machinery, but I've always supposed that we begin with the standard and look for corresponding features in the specimen. I'll describe it that way, but if there's any reason to turn it round, we can do so. (I've avoided the word "template" for that reason; this terminology seems to work not too badly.)

The standard and specimen images images, once loaded, are static; the only parts that change during the matching are numbers in the matching layer.

The matching layer is an array of processors, of the same dimensions as the images. Each processor corresponds directly to one pixel in the standard image, and its job is to find a corresponding location in the specimen. Apart from its knowledge of its own standard pixel, it has two sources of information : it is also aware of the states of *all* the retina pixels, and can therefore direct its attention to any part of the retina; and it can see what its neighbours are doing. (There is no reason why it can't also be aware of the states of further pixels of the pattern layer in the neighbourhood, but I haven't explored that avenue.) All processors execute the same programme. We imagine them to execute in parallel.

As well as the connections described above, each processor has an internal state which includes the coordinates of a point in the retina. (The coordinates are here assumed to be Cartesian, and labelled x and y, though other systems may be found useful and are obviously equally valid.) These are taken to be the coordinates of the part of the retinal image which corresponds to the point on the standard image corresponding to the processor. Rotations may also be included in the state if the matching error depends on direction.

CORRESPONDENCE.

What is a "corresponding" feature ? Ideally, The corresponding feature of a point in the standard image is the point in the specimen image which is related to the whole of the specimen picture in the same way as the original point is related to the standard picture. That's a clumsy way to put it, but the idea is that, if the two pictures are faces, the corresponding point of the tip of the nose in one picture is the tip of the nose in the other.

In effect, the function of the matching layer is to identify a "correspondence point" in the specimen for each pixel in the standard image – or, more precisely, for the central point in each pixel. If the standard and specimen images were identical and perfectly aligned, then the correspondence points of the standard image would be in exactly equivalent positions in the specimen, so (assuming that the layers are horizontal) a notional line from standard point to corresponding point would be vertical. Think of the matching layer as a sheet of plain glass, allowing the vertical "rays" from the standard to pass through without refraction. It seems reasonable to regard this as the base position, and to measure the positions of the correspondence points in distorted images by the north and east displacements from the standard positions. This convention has the significant advantage that it puts all processors on the same footing, so it's easier to write the same programme for each.

For a distorted specimen, think of the matching layer as a more or less distorted sheet of glass, refracting the downward "rays" from the standard in the direct of distortion of the specimen, so that they end up on the corresponding points of the specimen.

There is no reason why the correspondence points should coincide with the centres of the specimen pixels – indeed, with any distortion they are likely to move into intermediate positions. A simple example of this behaviour is the case in which the specimen is identical with the standard but scaled to half the (linear) size – then the correspondence points of up to three quarters of the standard points must lie between specimen pixels.

WHAT IS GOING TO CORRESPOND ?

Exactly how, or even whether, the method will work will depend on the nature of the scene. Here are some suggestions for different cases; I don't suppose the selection is exhaustive.

- For **line drawings**, only the lines are significant; the spaces are not. I'll always assume that the lines are black and the spaces are white – obviously this is purely convention. This is an important case, as it's more or less what you get from edge detection. It's also the obvious starting point for object identification, I think. Note that this is not at all the same thing as character recognition; characters are made of thick lines (except in very artificial experiments), while edges are computed from images and are only one pixel wide. (Otherwise they're not edges.)

- For **grey-scale images**, the problem is quite different. Here all pixels are significant, but it is not at all clear how to compare them. Simply using the given pixel intensities is doomed to failure, because a simple change in illumination of the specimen would cause a more or less massive change in the "error" even with the same scene. Differences in density are perhaps the obvious next possibility, but may not be the best. The scale still depends on the illumination and is therefore not a constant for the specimen scene – and may even vary significantly across the scene. It's worth a try, but density gradients sound somewhat more promising, and also indicate a direction which must be matched too. Some sort of intensity normalisation could be attempted before starting, but again doesn't take account of differences in illumination across the scene. As against that, grey-scale information can be informative about the shape of an object even when it doesn't have sharp corners, where edge detection is less effective.

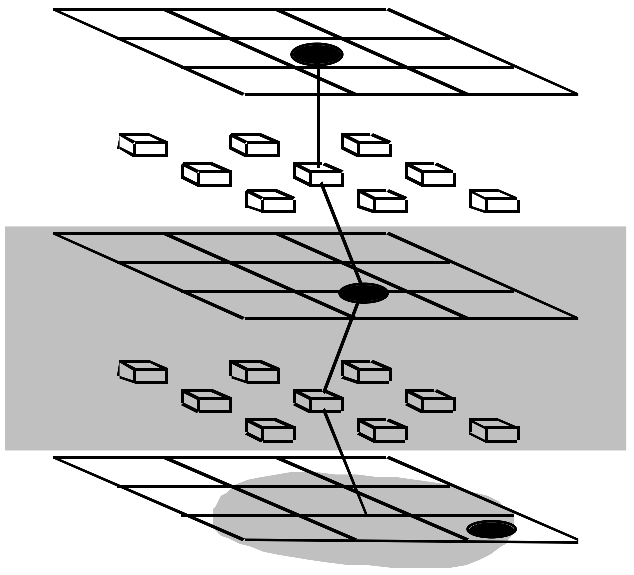
- **Coloured images** are even more complicated; there are three (at least) gradients to worry about, and they may not all point the same way. In many cases, too, the colour is irrelevant for recognition, which we want to work equally well for red cars and blue cars. Edge-detection is strongly indicated; note too that two similar objects of different colours are likely to appear with different intensities in grey-scale images, which doesn't make them easier to use.

- **Occluded images** introduce new problems. If we know about the occlusion, we should be able to link together visible portions by just observing distortions and ignoring pattern mismatch over the occluded area. (Ideally, we have another pattern to match in the occluded area, but that introduces all manner of exciting new possibilities; leave it for a while.) If we *don't* know about the occlusion, we should be able to infer it somehow from a region of totally haphazard pattern errors in a context of well matched pixels. If we can identify a connected field of such errors, we can try to guess about the occlusion. This is probably a function of a sort of monitor network rather than of the basic one. Look at some of Grossberg's work, perhaps.

PRINCIPLE.

My idea is to approach a correspondence iteratively by repeatedly adjusting an initially undistorted guess at the correspondence points. The aim is to match line drawings (grey scales are probably far too hard to interpret, and probably not very sensible anyway – see below), so the correspondence is a matter of matching black points on the standard to black points in the specimen in such a way that the distortion of the standard is minimised. The matching layer is not expected to deal with gross distortions, so the initial guesses will, in some cases at least, be not too far wrong. If the correspondence points at these positions can be moved a little towards what appears to be their correct positions, the match will be improved.

The process can conveniently be understood by considering an extension of the system described above produced by adding two further imaginary layers between the matching layer and the specimen.



In the diagram, the imaginary layers are shown on a grey background. The situation illustrated is an intermediate state in matching, where the only matchable items are the black spots in standard and specimen images. The imaginary grid shows the current distortion, resulting from previous iterations; the standard black spot is currently "believed" by its pixel's processor to match a somewhat displaced point, shown on the imaginary grid as lying on the line between pixels. The coordinates (more precisely, the displacements) of the displaced point are saved by the processor. This information is then used by the corresponding imaginary processor (in fact, a part of the "real" processor) to inspect a neighbourhood of the distorted position (shown as a grey area on the specimen grid) for potentially matching points. In this case, there is just one such point in the neighbourhood, which happens to be the point to be matched; in the next iteration, the displacement will be changed to move the displaced point towards the point to be matched.

In this operation, only the displacements of the black points will change, while all other correspondence points will be unaffected. The result amounts to a comparatively strong local distortion of the picture, with no change elsewhere; the improved match is gained at the cost of increased distortion. It is significant that the absence of change in the correspondence points of the "white" pixels does not arise from strong evidence that the points should not be moved, simply from an absence of evidence that they should. We may therefore permit the matching layer to adjust the coordinates in such a way as to reduce the distortion, and this is the next step of the process. This *relaxation* could also reduce the goodness of the match, but provided there is an overall improvement we are better off than when we started. These two steps are then alternately executed until the correspondence becomes stable. The intention is to simulate something like the effect of drawing standard and specimen images on rubber sheets, then moving the lines of the standard to coincide with lines on the specimen, while leaving the spaces unconstrained except for the forces from their neighbours.

It is convenient to think of these changes in terms of a metaphor of forces. We imagine that all black points in the standard are attracted to all black points in the image, and that there are elastic forces operating within the image to even out the local distortion. I shall use this metaphor extensively in the rest of my discussion.

I think that might be somewhat different from the model used by Mark Scaletti. From memory, I think he worked out all the "forces" and tried to combine them into one; that led him into various sorts of difficulty in determining the relative proportions of the forces, etc., and eventually to the definition of a third "viscous" force which complicated the computation without conspicuous benefits. Treating the two phases separately should achieve the same result, but make it a bit easier to implement.

Apart from some specific discontinuities (occlusions, etc.), the transformation should be "smooth" and continuous. A mathematical definition would probably say that any continuous line in the standard picture should be mapped (one-to-one) onto a continuous line in the specimen picture, though without requiring that the lines should be the same shape, and that if any points A, B, and C are in that order on the standard line then their images are in the same order in the specimen line. One might also have to require continuous derivatives. I hope I won't need a rigorous definition; the operation is rather commonsensical, and I should be able to muddle through. (Though a more precise definition would do no harm.)

The result of the distortion should be the "best" mapping of standard onto specimen. "Best" includes components from the goodness of the match between standard and specimen, and from the magnitude of the local distortion. Translation, rotation, and scaling the standard as a whole doesn't count as distortion.

There is some balance between the severity of distortion and the goodness of match. I don't know where it is.

ALGORITHM.

```
Main :
    Read standard image;
    Read specimen image;
    Match;
    Report results.

Match :
    Repeat until happy :
        Distort;
        Relax.

Distort :
    For each processor :
        Find local match;
        Evaluate distortion force.
    For each processor :
        Move the correspondence point.

Relax :
    For each processor :
        Evaluate relaxation force.
    For each processor :
        Move the correspondence point.
```

Details of the elementary procedures remain to be determined, but I think that once the skeleton is there it should be fairly straightforward to experiment.

Possible distortion algorithms.

It's easy to imagine distortion algorithms of arbitrary complexity. It's also unwise. That's what Mark did, and it led him into all sorts of complications. I think a good first try would be to restrict the view of each processor to the specimen points within some quite small radius of the current estimate of the correspondence point – perhaps to the closest specimen point and the surrounding eight points. If that doesn't work, we can try something else, but I think it should perform provided that the distortion is not too big.

Matching only applies to correspondence points of black pixels in the standard image. (That isn't a law of nature, but it seems to fit in with how I feel that I match pictures. Totally unreliable evidence ! – but I present a sort of justification below.) If a black correspondence point is in a black specimen region, there's no distortion force; it's in a black region if the nearest pixel centre is black. (That means that there's no arbitrary tendency to force correspondence points to lie on specimen points.)

A black correspondence point lying in a white specimen area is attracted towards any black specimen points which it can see. Again, there's no law of nature, but a vector attraction decreasing with increasing distance seems appropriate. The resultant "force" on the correspondence point determines its direction of motion, and the distance moved is some multiple of the magnitude of the force.

Bearing in mind that we're not trying to handle massive distortions, limiting the range of the attraction makes sense. If there isn't a match for some feature, it would be better to leave it dangling than force it to identify with something quite inappropriate.

Possible relaxation algorithms.

We want two things out of the relaxation measurement : we want a figure which gives us a reasonable measure of the distortion, and which we can use to tell when the system has settled down, and we want some measure of the distortion around each correspondence point so that we can tell which way to move it, and how far, in order to relax the distortion. That means that we have to define the place at which the correspondence point "should" be to minimise the distortion.

I think that the simplest approach is to guess that the relaxed position of each correspondence point is the average position in the current distorted image (the imaginary layer of the figure above) of the four neighbouring points which correspond to the neighbours in the cardinal directions in the standard image :

$$(x_{im}, y_{im}),\ \text{where } x_{im} = (\Sigma_{n \quad \text{neighbours}(i)} x_n) / 4,$$

and y_{im} is defined analogously. (This assumes a square grid; if you want a triangular, hexagonal, or random grid, work it out for yourself.) The relaxation force can then be taken as some function of the vector from the current displaced position to that relaxed position. For small distortions, it probably doesn't matter what function you use provided that it's plausible; for large distortions, it probably doesn't matter at all, because if the accuracy of the results isn't determined by small distortions it's probably no use anyway. It is necessary, though, that the force be big enough to move a black point away from a black neighbourhood if it's wrongly matched. A force which increases with distance is appropriate (and conveniently fits in with the elastic sheet image).

That doesn't work at the edges. The edges must be free to move (because a bodily shift of the image isn't a distortion), but the simple average as defined above is obviously biased. (Mark said he allowed the edges to move, but it isn't obvious from the diagrams in his thesis.) I think that the easiest way to deal with it is to imagine phantom neighbours which replace those which are "missing", and are always placed in a suitable direction from the unbalanced point at a suitable grid distance. It might be appropriate to place the perimeter points so that the penperimetral points are at their equilibrium positions as determined by the relaxation calculations. Only this position guarantees that the system will be stable if there is no relaxation force in the rest of the network. (I think.)

INTERPOLATION, OR SOMETHING OF THE SORT.

I don't see how one might try to prove it, but I think that the simple models of the process in terms of forces ought to work smoothly if the specimen image is continuous – which, of course, it isn't. With a discrete specimen image, one might foresee problems, and the algorithms should be designed to cope with them.

For example, in the ideal case, a perfect black-to-black match need not anchor a pixel, provided that (almost) equally good matches are accessible; only serious (which remains to be quantified) decreases in goodness of match should prevent possible moves. A matched black point should therefore be able to move freely along a black line if this improves the quality of the solution by decreasing the overall distortion.

In practice, with the obvious simple discrete representation of a line drawing, black lines are not continuous, but a series of steps. How can one compute the distortion forces to achieve a reasonable approximation to the ideal case ?

There are several obvious ways to attempt this, but all those I can think of are sub-perfect. Pixel intensities (or whatever) might be regarded as concentrated at the centres of the pixels, but then even points displaced onto the line will still experience strong distortion forces tending to move them to the pixel centres. Another view is that once a black point has been displaced into a black pixel it is home, and experiences no distortion force, but that could easily lead to oscillations across pixel boundaries with shifts caused alternately by distortion and relaxation. Either of those views deals badly with events at a step in the line, where the line itself – purely as an artefact of the representation – appears to the algorithm to be discontinuous. Perhaps this could be addressed by a thickening algorithm, which ensures that lines are never thinned to vanishing point, but as in practice the lines are likely to come from an edge-detection algorithm and are quite likely to be discontinuous anyway, that isn't obviously the optimum solution; we really do need an algorithm which will match a continuous standard line to a broken and approximate specimen line without complaining unduly.

DOING THE CALCULATION.

In some of my earlier notes, I went into some detail about the calculation; that was mainly just to show that it could be done somehow. Now I'd rather not be too specific, because I suspect that a certain amount of experimentation will be necessary. This discussion is therefore much more general.

I think it's possible to lay down a few (two ?) principles. These are open to debate, but both are based on considerations which should be taken into account somehow.

- The first principle is that it makes a lot of sense for the processors to act using only local information; global error minimisation is probably a bad idea. That's because it's likely that the degree of distortion will vary significantly from point to point in the specimen image, and the system will have to rely on any local good correspondences which it can find to guide it in resolving the other bits; a processor which can see a good local match should therefore go for it. (In an earlier note, I proposed a global "steepest descent" method; I now think that was a mistake.)

- The second principle is that the movements at any step must be restricted. It is important to retain the continuity of the mapping from standard to specimen, and large movements risk breaking this by (for example) reversing the order of two black pixels on a line. A good way to preserve the continuity is to avoid breaking it. This suggests that the distortion should be managed as a series of small adjustments. How small ? I would guess that a maximum shift of the order of half a pixel at each stage would be appropriate. This guarantees that any tendency to "twist" the correspondence can be found and stamped out before it happens.

MEASURING DISTORTION.

The distortion of the network is measured solely in terms of the retinal coordinates recorded by the matching neurons. Following the notion of distortion introduced above during the discussion of the relaxation step, each neuron compares its coordinates with those of its neighbour, and records a distortion error if its x and y coordinates deviate from the arithmetic mean of the neighbours' coordinates. The seriousness of the error is regarded as increasing as the distortion increases, and can sensibly be measured by the square of the distortion. The contribution of point i to the distortion is :

$$E_{id} = (x_i - x_{im})^2 + (y_i - y_{im})^2.$$

Notice that this function records no distortion for translation, scaling along either coordinate, shear, or rotation. That may be deemed rather too liberal. Translation, rotation, and isotropic scaling can certainly justifiably be regarded as distortionless, but the others are less easy to accept. Alternative ways of working out the error are easy enough to develop, but fussier and harder to write down.

There is nothing specially miraculous about the squares of the displacements. If we just want a number, I don't think it matters much whether we use that or the sum of the magnitudes of the displacements or something quite different; the sum of squares is probably preferable if we're ever going to go on to clever mathematics (steepest descent methods, etc., which we're not), because it's smoother and has a nice minimum at the optimum point, but (unless you're eager to get into mathematics) there's not much urgency about that. The squares weight big displacements more, so that measure will favour a lot of little displacements over a few big ones – I think that's what we want, but argument is welcome.

What do we do with black pixels that aren't matched ? Nothing, I suspect. If we're looking for a number to minimise, the contribution of an unmatched pixel is not obviously well defined, but whatever it is it won't change the quality of the successfully matched points.

HOW GOOD IS THE RESULT ?

On the other hand, if we want some figure to represent the quality of the result, we have to take into account both the distortion and the degree to which the pixels are matched.

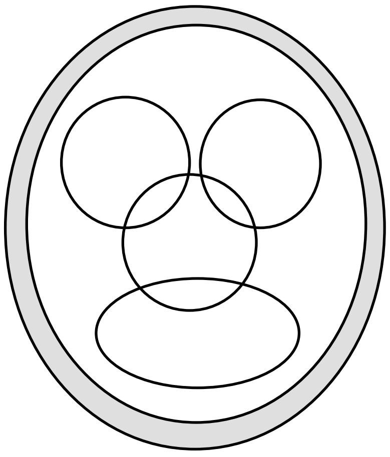
Do we need such a figure ? I don't think that the algorithm needs it. It has to have a criterion for stopping ("Repeat until happy :"), but the rate of change of the total distortion might be sufficient for that. (Note that the *value* of the distortion won't; it starts off at zero, which is the "desired" result.) Like a smooth distortion measure, we only need a defined measure of quality if we want to do mathematics.

I think it would be sensible not to worry about it too much, but see the identification of a useful measure by experimentation as an interesting result of the project. Then someone else can do the mathematics.

NOTES.

Unmatched features : We might expect that, usually, the specimen to be quite a good match to the standard. That's because this bit-by-bit matching is intended only as the bottom level of a more elaborate scheme, with the higher levels selecting features rather than trying to match bits.

In such a system, a (probably grossly oversimplified) face recogniser would have a high level representation of a face something like this :



This specifies only that the face will have an edge somewhere in or near the grey area, and that two eye features, a nose feature, and a mouth feature will usually be found somewhere in or near the regions outlined. A more sophisticated model could also have optional features for spectacles, moustache, etc. Subsystems for each feature would attempt more detailed matching (probably again using more than one level), generally with a variety of standard patterns – open eyes, closed eyes, happy mouth, sad mouth, etc. Putting all this together, the pixel matching shouldn't have to deal with any drastic distortions except by giving up.

The "recognition" system works by an ART-like matching of input pattern to known templates, with the best match chosen as that to be accepted.

Realism : The main obstacle to realistic simulations is the time taken to process large bit images; that's why so many people use little ones. Unfortunately, little ones are unrealistic in many ways. The smallest distortion you can make is comparatively large; a straight line rotated is either grossly unstraight or rotated through a large angle.

This is a particularly severe problem with lines. A line drawing in a small picture can hardly use lines thicker than one pixel, but these are very sensitive to noise. In real vision, lines are a few pixels wide, and much harder to break with the odd noisy pixel; with thin lines, a little noise can break continuity and produce disjoint pictures. This shouldn't worry us directly, but with thin lines almost all the black points will have to be dragged into a fairly precise position, while with thicker lines black points within the lines will be able to relax freely.

(Thinking of edges rather than lines is perhaps better; edges are clearly one pixel wide, and if we always know that then perhaps we can adapt to it.)

Big shifts with no distortion : The requirement that a copy of the standard which has been merely translated or rotated should not be regarded as distorted implies a degree of acrobaticity in the matching processors. If I want to recognise something which has been moved or turned round, I may well move my eyes or tilt my head to bring the image to a position and orientation on my retina at which I find it more familiar. The distortion network can't move its head, so has to compensate by relying on tricky processors to do the job for it.

An interesting possibility for exploration might be to use this property as a way of tracking a moving object in a scene. Once a match is found, it should be "easy" to keep it up to date by successive adjustments of the displacements as the object moves.