University of Southampton

Extraction of arbitrarily moving arbitrary shapes by evidence gathering

by

Michael G. Grant

A thesis submitted for the degree of
Doctor of Philosophy

in the
Faculty of Engineering and Applied Science
Department of Electronics and Computer Science

February 2002

Abstract

There are currently available many approaches aimed at tracking objects moving in sequences of images. These approaches can suffer in occlusion and noise, and often require initialisation. These factors can be handled by techniques that extract objects from image sequences, especially when phrased in terms of evidence gathering. As yet, the newer approaches to arbitrary shape extraction avoid discretisation affects but do not include motion. The moving-object evidence gathering approach has yet to include arbitrary shapes and can require high order description for complex motions.

Since the template approach is proven for arbitrary shapes, we re-deploy it for moving arbitrary shapes, but in a way aimed to avoid discretisation problems. As the template approach has already been seen to reduce computational demand in the extraction of arbitrary shapes, we further deploy it to describe the motion of moving arbitrary shapes. As with the shape templates, we use Fourier descriptors for the motion templates, yielding an integrated framework for the representation of shape and motion. This prior specification of motion avoids the need to use an expensive parametric model to capture data that is already known. Furthermore, as the complexity of motion increases, a parametric model would require increasingly more parameters, leading to a rapid and catastrophic increase in computational requirements, whilst the cost and complexity of the motion template model is unchanged. The new approach combining moving arbitrary shape description with motion templates permits us to achieve the objective of low dimensionality extraction of arbitrarily moving arbitrary shapes with performance advantage as reflected by the results this new technique can achieve.

Acknowledgements

I would like to recognize and thank my supervisor, Mark Nixon, for his guidance throughout the research in this thesis. I am also grateful for the many new words I have learned from him ("adumbrated" for example and, most especially, that word beginning with "o").

I would also like to acknowledge the invaluable assistance and support from friends within ISIS. In particular, I appreciate the help from Jamie (who actually managed to read all of this without falling asleep too often) and also Karl and Richard (no GW!) for their discussions and help on subjects of marginal value to them. I also have a few words for Jamie and Jaz: sewer, hoarse, Rutherford, hyperventilation and aspirator. I look forward to matriculating with you...

Thanks are also due to my friends outside work for the diversions and amusements that make life more interesting.

Finally, I dedicate this work to my parents for all their efforts and support throughout the years, and for the upbringing that encouraged me to come this far.

Table of Contents

List of Figures

List of Tables

Nomenclature

Symbol	Meaning
$c_x(s)$ and $c_y(s)$	A shape-template curve to be described by FDs
A_{xk}, b_{xk}	Fourier Descriptors of x component of $c_x(s)$
$\overline{FD}_x$	Vector of FD harmonics (x part)
$\upsilon_x(s,\overline{FD}_x)$	x component of curve reconstructed from $\overline{FD}_x$
$\boldsymbol{a}_s$	Vector of shape scale and rotation
$R_x(s,\boldsymbol{a}_s)$	Scaled and rotated reconstructed curve
T_{ref}	A temporal co-ordinate in a sequence (e.g. frame number)
$\overline{\lambda}(P,T_{ref})$	Image sequence, P is a co-ordinate in the spatial image
$\overline{\omega}(s,T_{ref},l,\rho,v_x,v_y)$	Accumulator vote pattern, with various parameters including velocity v_x and v_y,
$A_{P,T_{ref}}$	Accumulator vote pattern, offset from an image co-ordinate
$M\ (\overline{c},\overline{d})$	Matching function – defines how votes are placed in an accumulator $\overline{c}$, according to a point in the vote-pattern set $\overline{d}$
$MT_x(T_{ref})$ and $MT_y(T_{ref})$	A motion template curve to be described
$\overline{MTFD}_x$	Vector of FD harmonics (x part) describing a motion template
$m_x(T_{ref},\overline{MTFD}_x)$	x component of reconstructed motion template
$\xi_x(s,T_{ref},T_{off},\mathbf{a}_s)$	Reconstructed shape template with motion template-derived time-dependent rotation and scaling factors
$\boldsymbol{a}_m$	Vector of motion template rotation and scale – transforms the motion template globally
$\mu_x(s,T_{ref},T_{off},\mathbf{a}_m,\mathbf{a}_s)$	Reconstructed shape template, motion compensated for time T_{ref}
$\overline{\omega}(s,T_{ref},T_{off},l_T,\mathbf{a}_m,\mathbf{a}_s)$	Motion template vote pattern

1 General introduction and motivation

In recent years, the primary application of basic computer vision research in Southampton has been gait recognition. The problem has provided a spur to develop new technique in fields ranging from statistical description to feature extraction. This thesis concerns developments of the latter approach - that is, in generalised feature extraction. Although the work here is generic (in terms of moving shape analysis), we use gait recognition as the exemplar and stimulus.

In order to be able to recognise people, knowing their approximate location is a prerequisite. When considering a video sequence, as required for motion-based recognition, it also becomes necessary to locate them in each frame. Current approaches to this problem tend to rely on tracking techniques - an object is located in one frame and followed or tracked in successive ones. Whilst these methods generally permit real-time implementation, they depend on good definition of the target in the current or recent frames and an appropriate initialisation. In noisy or occluded imagery, as is common with complex scenes, a substantial number of frames may be corrupted or unusable leading the tracking method to lose the target and perform non-optimally. Naturally, poor initialisation also easily leads to apparent failure.

So, tracking techniques have a number of negative characteristics in addition to their positive ones - particularly that they do not consider a video sequence as a whole, but as a linear series of images. There are obvious benefits that arise from a more holistic approach; especially that correlation across a sequence can be examined. In most cases changes happen slowly, a fact exploited in motion encoding (e.g. the MPEG suite [64]). Slow changes imply strong correlation between nearby frames or even over many frames.

Few algorithms in computer vision use temporal correlation across a sequence to improve feature extraction. One of these is based around the Hough Transform (HT), giving it the strong theoretical grounding and robustness enjoyed by evidence-gathering methods. The Velocity Hough Transform (VHT) allows sequence-based extraction of conic sections that are moving in a parametrically described manner (for example, linear or sinusoidal motion).

Although the VHT was used previously to locate the leg of a walker, the shape model (a line) and motion model (sinusoidally bobbing linear motion) used were only adequate for the limited experiments possible at the time. The nature of the VHT

provided two barriers to accurate modelling - the inherently restricted generality of its shape and motion models.

The first barrier to using the VHT for person location is that body shape is not well represented by conic sections. Limitations of the shape description in the VHT preclude the more complex and even arbitrary shapes that are required to give a reasonable reproduction of a human shape, without excessive computational resources. Hence, the first part of our work was to extend the VHT to allow arbitrary shape extraction, to give a continuous-template variant of the VHT (CVHT). We use Fourier-Descriptor templates to achieve efficient arbitrary shape representation.

The second barrier is that people do not move in a simple parametric way but rather in a complex and situation dependent way. The motion description in both the VHT and CVHT suffers from the same drawbacks as the original shape description in the VHT - a lack of sufficient generality at a supportable level of computational resource usage. In the second part of our work, we alleviate this restriction also. Again, we use Fourier-Descriptor templates and thus gain a consistent framework across both shape and motion description.

In this thesis, we present two novel developments of the evidence-gathering paradigm that, together, allow for efficient and robust extraction of arbitrarily moving arbitrary shapes. The following chapter details related work and also the foundations of the new developments. It ends with a more detailed discussion of the contributions to the field. We continue by describing our approach to arbitrary shape extraction with parametric motion (Chapter 3) and with arbitrary motion (Chapter 4). In each section and for each new technique, we discuss the issues, present theory and examine results of some comparative evaluation and performance analysis. Finally, we give conclusions and suggest directions for future research, presenting some initial work on these directions.

1.1 Publications related to this work:

There are currently four publications associated with this thesis. These are:

- [20] A poster at the IEE colloquium in London (1999) that described early work on the CVHT
- [21] An oral presentation at BMVC99 that gave a full analysis of the CVHT and mentioned early work on motion templates
- [22] A poster at BMVC2000 that presented the developed motion template work
- [23] A paper in Pattern Recognition that contains a complete, but excerpted, version of this thesis, covering the development of both CVHT and motion templates. Appendix 10 contains the final version of this paper.

2 Related work and foundations

This chapter begins by setting the historical context of the basic technique underlying this novel work - the Hough Transform. We continue by detailing the immediate foundations of the new developments and, where appropriate, indicating related and relevant work. The material in this chapter concentrates on the major developments that have direct relevance to this work – for more general reviews of the extensive literature relating to the HT, see the surveys by Leavers [38] or Illingworth [30] or books with major sections on the subject [15, 46].

The Hough transform [26] was originally formulated to detect lines in an image – its first application was to automatically detect tracks from pictures of bubble chambers. The implementation used the slope-intercept parameterisation of a line, which has an unbounded parameter space since the parameters can have an infinite range. The algorithm was later introduced to the computer vision research community [53]. Its principal advantage is that it produces optimal results since it is an efficient form of template matching [57], which is optimal in Gaussian noise.

The slope-intercept formulation's unbounded parameter space made the HT for lines impractical for general scenarios until the transform was extended [17] to use the normal parameterisation. This adjustment puts bounds on the maximum size of the parameter space and makes line detection using the HT technique viable. The two parameters are constrained: the angle component's range is limited to $\pm\pi$ radians and the distance component is restricted to the length of the image diagonal in pixels.

In the same paper, Duda and Hart also described how the HT algorithm could be modified to detect any analytically defined shape, showing the example of a circle in detail. Following this, the HT was extended to conic sections (circles [33] and ellipses [58]). These extensions were possible and fairly simple because the essence of the HT algorithm is to match feature points (e.g. edge pixels, vertices or depth values) to parameters of a constraining equation - not just to match pixels to a line's parameters. So, the constraining equation can be changed from a description of a line to that of a circle or a more complex shape. A formalisation of the HT [52], which brings out these properties, is adumbrated in Appendix 8.1.

Early adaptations of the standard HT increased the complexity of the constraining shape equation and thus the dimensionality of the HT – lines require a 2D parameter

space, circles a 3D space, ellipses a 5D space, etc. Extrapolation suggests that for a parametrically defined arbitrary shape (effectively a high- or infinite-order polynomial) the standard approach would require a nearly infinite dimensional parameter space. In contrast with such an approach [37], which required an exorbitant accumulator space, an arbitrary shape HT actually only needs to accumulate for the (relatively few) appearance parameters, provided that the shape to be located is already specified [4]. Instead of searching for the best fit to the parameters of an arbitrary polynomial, the only parameters that need examination are those that tug and stretch a template shape until it matches the target – such as position, rotation and scale. We will later see how templates can be used efficaciously not only in shape extraction, but also in motion extraction and description.

2.1 Generalised Hough Transform

Merlin and Farber [39] first considered general-shape detection using the HT but their method provided no means for detecting rotated or scaled shapes. Ballard developed the full mapping [8] for arbitrary shapes with rotation and scale invariance – the Generalised Hough Transform (GHT). The GHT replaces the analytic parametric constraints in the HT with a non-analytic tabular representation of an arbitrary shape. This table (the "R-table") describes the position of feature points in the template, or target, shape relative to a reference point and is indexed by the gradient direction information at each feature point. Compared with Merlin and Farber's method, this table also increases the efficiency of the algorithm by reducing the number of feature points under consideration to those that fit the additional gradient direction constraint. Merlin and Farber trace entire instances of the template shape in the accumulator, whereas the GHT only adds particular points from the template contour to the accumulator. In the GHT, the lower number of votes cast reduces the amount of noise in the accumulator generated by false votes (provided that the gradient data is of good quality [7]) and can also improve the computational speed.

2.2 Fourier-descriptor template representation

Discretisation or quantisation errors are one of the areas of the HT that has been frequently researched, producing analyses of the source of the error, its consequences and algorithms to minimise or remove it [34, 47, 59]. However, the GHT in particular introduces an additional source of error – the non-analytic shape representation. The problems with the GHT's R-table representation have been described in the literature, most recently and in greatest detail in [5] although Grimson's work is better known [24]. They essentially derive from the fact that it is a discrete representation sampled at a particular resolution. When the template is scaled or rotated, there can be problems with aliasing and rounding errors. Figure 1 shows the effects of scaling and rotating the discrete set of points comprising the original shape (Figure 1a, shown as scaled by 1.0, with no rotation). Figure 1b-d shows the set of points at different orientations, whilst Figure 1e-h and Figure 1i-l repeat the rotations at two larger scales, double and quadruple respectively. Clearly, there is missing data in the new sets of points - the points in the original have become separated due to inadequate sampling at the new scale. Furthermore, the effects of discretisation are particularly evident in the rotated instances where points have moved from their true position due to rounding errors. If the shape had been shrunk, the points in the original would merge, effectively oversampling the shape. Given higher (or multiple) resolutions this error can be minimised, but it will always be present. Also, additional/alternative resolutions are frequently unavailable.

Distortions are inevitable when working with discrete systems. Nevertheless, the worst effects can be mitigated by maintaining a continuous representation for as much of the process as possible. Using an analytically defined curve as the template representation makes it possible to defer discretisation until after rotation and scaling. Elliptic Fourier Descriptors (FDs) [35] have been deployed in an adaptation of the GHT [4] to give such a continuous representation. Instead of recovering vote co-ordinates from an R-table, they are instead calculated from the FDs. This avoids the extra quantisation step inserted by the GHT (discretisation occurs in template transformations and again in the accumulation phase), thereby restoring the robustness of the original, analytic, HT formulation. In support of this, Aguado [5] found that extraction using FDs was possible with greater than 90% noise/signal (the

ratio of the number of image points to the number of points that define the shape) whilst the GHT generated prominent false peaks for values close to 70%.

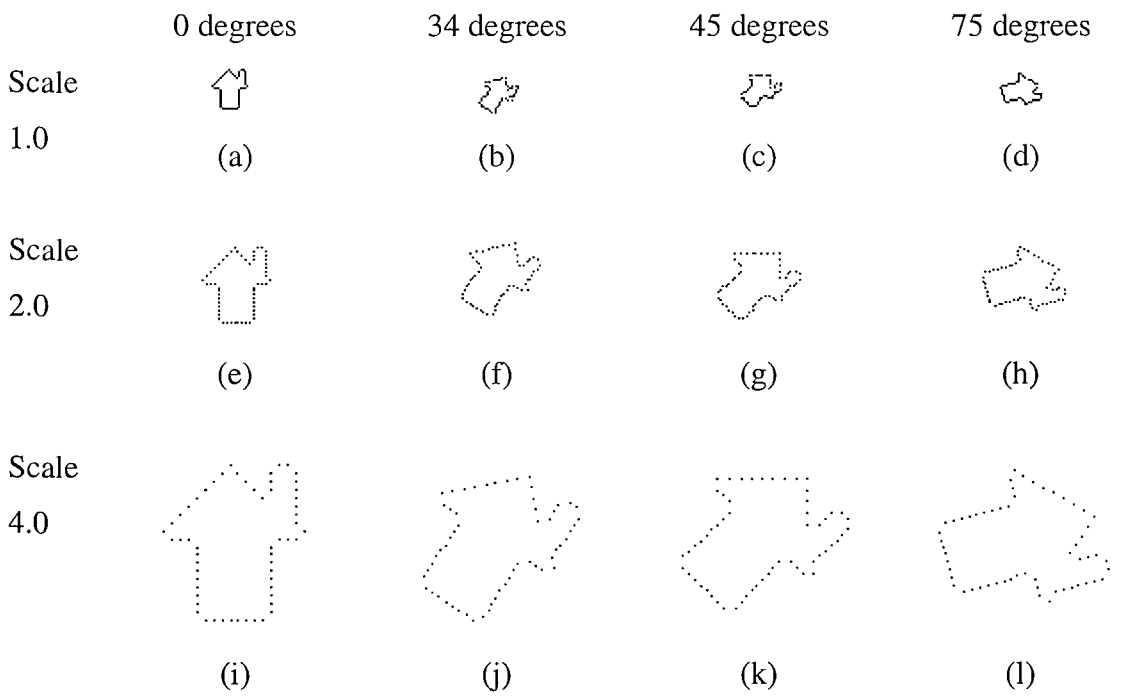


Figure 1: Discrete shape representation (GHT) at varying scales and orientations

The FDs are primarily used to provide a continuous representation of the original template shape but there are other benefits. For example, they also serve to in-fill between points from the original specification of the template, effectively performing an interpolating function when oversampling. Clearly, if the original template is at a smaller scale than the reconstructed one, there will be no additional detail provided by the FDs, just interpolation of the initial data. Interestingly, selecting particular bands of harmonics (thus under-sampling the FDs) can extract "significant" natural scales (i.e. a fundamental part of the structure) from a model [54]. Figure 2 gives Fourier described samples of the shape in Figure 1. Here, a continuous set of points is derived at different scales and orientations, and remains as contiguous and continuous as the original shape. Note that truncation of the Fourier series leads to rounding effects in regions of high curvature, a factor more noticeable at increased scale. However, in the figure, most of the rounded corners are artefacts resulting from the generation of the FDs from an eight-neighbour chain code (giving diagonals rather than right-angles at the corners). Other benefits of Elliptic Fourier Descriptors are their completeness, simple geometric interpretation, access to frequency information and the fact that they

can be easily produced from a chain code of the contour. For all these reasons they were suitable for this work. However, other analytic representations could equally have been used (e.g. cubic B-splines as in [61]).

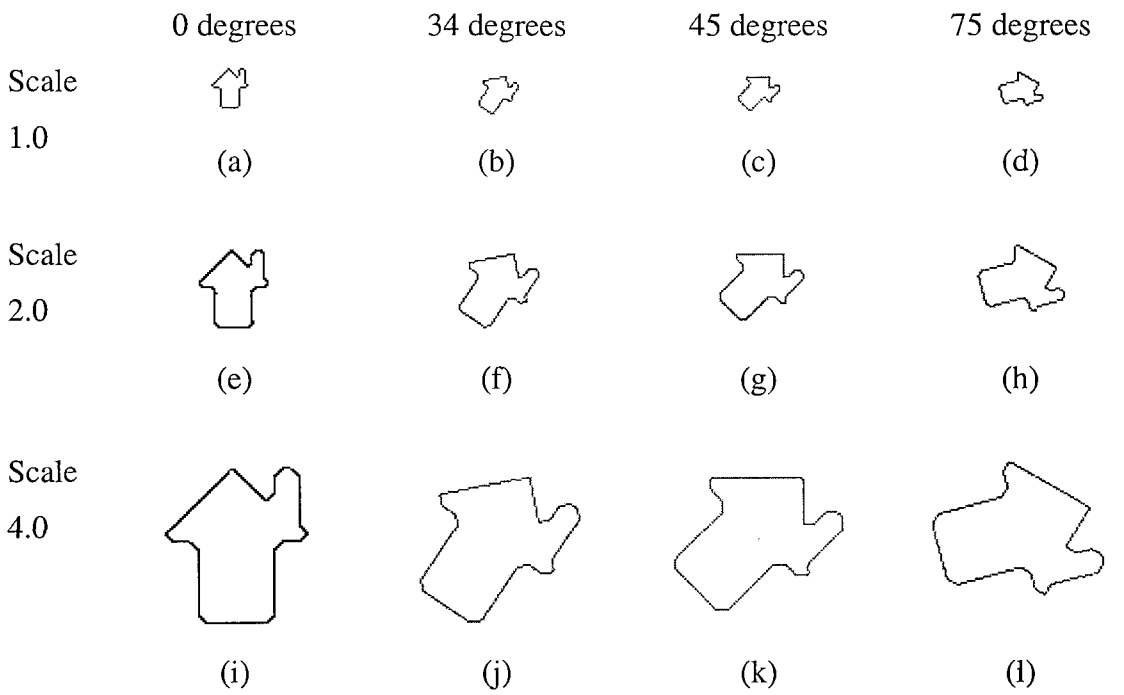


Figure 2: Continuous shape representation (FDs) at varying scales and orientations

Elliptic Fourier Descriptors encode a contour as two Fourier decompositions by considering the x and y components separately. The decompositions contain a number of frequencies limited by the number of harmonics used. The two Fourier components can be recombined and viewed as a series of linked, rotating ellipses - one for each harmonic frequency. The speed of rotation is defined by this frequency, the size of the ellipse by the maximum magnitudes of each Fourier component and the initial orientation of the long axis by the relation between the components.

One important implementation concern that quickly arises is the number of harmonics to use when representing a shape. Evidently, the more harmonics used, the better the reconstruction will be. In most cases, this is an asymptotic improvement and a balance must be struck between accuracy, computational costs and the requirements of the application. Here, we are using the FDs in the context of digital (thus discrete) image processing and hence there will be a limit defined by the pixel granularity and scale beyond which additional harmonics have no perceivable effect. This limit is dependent on the fineness of the pixel grid and the complexity of the shape

represented. Kuhl [35] specifies a simple procedure for analysing the maximum error of a representation from the original for a given number of harmonics, thus providing means of determining the number of harmonics for a given error automatically.

Another initial concern with FDs was that of disjoint or non-contiguous contours (i.e. one with gaps), since the method requires complete, cyclic curves. In the simplest case, that of a non-cyclic open curve, the recommended procedure is to loop back along the curve when one end is reached thus producing a full cyclic curve. However, the discontinuities at the end of the open curve mandate use of a larger number of descriptors than is usually required for closed shapes and this method fails in more complex cases with multiple breaks in the contour. A more general solution is to generate FDs that produce a shape without gaps and then use a masking function that causes the regeneration phase to ignore points that would fill in the gaps.

2.2.1 Theory

A curve defined by two sets of orthogonal co-ordinates, $c_x(s)$ and $c_y(s)$, parameterised by $s \in [0,2\pi)$ has elliptic Fourier Descriptors as follows:

$$a_{xk} = \frac{1}{\pi}\int_{-\pi}^{\pi} c_x(s)\cos(ks)\mathrm{d}s \quad \text{and} \quad b_{xk} = \frac{1}{\pi}\int_{-\pi}^{\pi} c_x(s)\sin(ks)\mathrm{d}s \tag{1}$$

with a similar equation for the y descriptors, where k is the harmonic number. The range of k defines the number of ellipses used to represent a model shape and thus how accurate the shape representation is (up to the practical limits mentioned above).

To reconstruct the original shape from the FDs, they must be converted from the frequency domain to vectors in the spatial domain. n FDs can be converted to vectors (along x- and y-axes) from the origin to a point on the curve by:

$$v_x(s,\overline{FD}_x) = \sum_{k=0}^{n}\left(a_{xk}\cos(ks) + b_{xk}\sin(ks)\right) \tag{2}$$

where $\overline{FD}_x = \{a_{x1}, b_{x1}, a_{x2}, b_{x2}, \ldots, a_{xn}, b_{xn}\}$, with complementary equations in y. The DC bias can be omitted by not summing the DC ($k=0$) terms. If they are included, they translate the origin so that it is the same as that used in the co-ordinate frame when the curve was originally sampled into FDs. When using chain codes, as we are, the origin is the start point of the chain. It is more intuitive in implementation (and has no appreciable effect on the HT) to omit the DC bias, moving the origin of a reconstructed shape template to the centre of mass. This tends to be more central in

the template than the start point of its chain code and thus makes it easier to visually interpret planes of an accumulator because peaks will be near the centre of the target.

2.3 Temporal evidence gathering

There is a substantial literature on the problem of tracking shapes in a sequence, for example the tracking of humans has recently been admirably surveyed [2]. Other surveys of interest include [1, 12, 18, 28, 40]. The surveys listed all deal with motion and thus, by implication, sequences of images. It is well known that most image sequences contain significant correlation across many frames - a fact commonly utilised by machine vision and video compression algorithms amongst many others. Earlier research has ranged from optic flow [32] to Kalman filters [10] and includes temporal templates, a form of which we use later in this thesis. Although much work has been done with spatio-temporal structures, the first evidence-gathering based technique to exploit this correlation for concurrent structure and motion analysis was the VHT [43]. The original implementation of the VHT extracted the optimum parameters describing a conic section moving with linear velocity. With simple extensions, it handles a subset of rigid motions that can be described parametrically. Hence, the *nature* of both the shape and motion is known *a priori* - a fact that is important for later chapters. To take advantage of the inter-frame correlation, the VHT collects evidence from the whole sequence into a single accumulator, concurrently extracting optimal structural and motion parameters. As a consequence of the additional information collated, the VHT appears to be more robust than a standard frame-by-frame tracking implementation, especially when the target is occluded or noisy. Any missing or damaged structural information in one frame can be compensated for by redundant data in others (for example, structural information from the target shape is often repeated in each frame).

Due to the global scope of the VHT, there is no need to initialise the algorithm to search in a specific area (although limiting the extent of the search is a possible optimisation). Another common motion estimation problem avoided by the VHT is that of correspondence. Points in different frames do not need to be matched since all the possible correspondences are examined implicitly in the accumulation phase. By the nature of evidence gathering, the best correspondences produce the highest accumulator peaks.

The motion model is parametric and thus can be extended from linear velocity by including extra terms. In this respect, an extension to the VHT [44] that found walking subjects using an articulation model required thirteen parameters. As illustrated by this example, the major disadvantage of the parametric motion model is that any extension increases the dimensionality of the accumulator and, thus, the computational resources required. In summary, the VHT enables the use of temporal correlation in an evidence-gathering framework, resulting in a powerful and robust extraction algorithm. Unfortunately, the modelling of both shape and motion is seriously restricted.

2.3.1 Implementing the Velocity HT

The VHT was originally formulated as an extension to the HT for circles [17], adding a velocity component and extending the single input image to a sequence. When casting votes in the accumulator, the HT for circles uses the following formulae as a part of its kernel:

$$\begin{aligned} a_x &= c_x + r \cdot \cos(\theta) \\ a_y &= c_y + r \cdot \sin(\theta) \end{aligned} \tag{3}$$

where a_x and a_y are the co-ordinates of a vote in the accumulator, c_x, and c_y are centre co-ordinates of the circle to be drawn in the accumulator (i.e. the co-ordinates of the feature point, e.g. an edge pixel). r is the radius of the circle being searched for and θ is the polar parameter of the circle. The accumulator is three-dimensional and stores a_x, a_y, and r. To extract the parameters of a circle in linear motion, the equations describing the points in a circle need only include a linear motion term, as:

$$\begin{aligned} a_x &= c_x + r \cdot \cos(\theta) + v_x \cdot t \\ a_y &= c_y + r \cdot \sin(\theta) + v_y \cdot t \end{aligned} \tag{4}$$

where c_x and c_y are the circle's centre co-ordinates, v_x and v_y are the velocities of the circle along the x- and y-axes, and t is the time reference of the frame relative to time $t = 0$ (normally the initial frame). These equations calculate the vote co-ordinates in an accumulator (now five-dimensional and storing a_x, a_y, r, v_x and v_y) for a feature point in a given frame of a sequence. After processing, the highest peak in the accumulator represents the best estimates of the moving circle's parameters and its centre co-ordinates at time $t = 0$.

The time reference of each frame is used to calculate the offset back along the path of motion for each vote. For example, if a circle moves forward at 1 pixel per frame, at time $t=3$ the circle has moved forward 3 pixels. In order to focus all the votes for the circle onto one peak, votes in the accumulator plane representing a velocity of 1 pixel/frame should be moved 3 pixels back to compensate. The velocity compensation is illustrated for a moving circle in Figure 3, which depicts the three-dimensional accumulator space x, y, v_x. Votes are drawn, centred on an edge pixel co-ordinate at several different frame times (shown as crosses). They are adjusted for the x velocity that is relevant to the plane of the accumulator in which they appear. The topmost set of circles has the correct velocity because the circles of votes all intersect at a point.

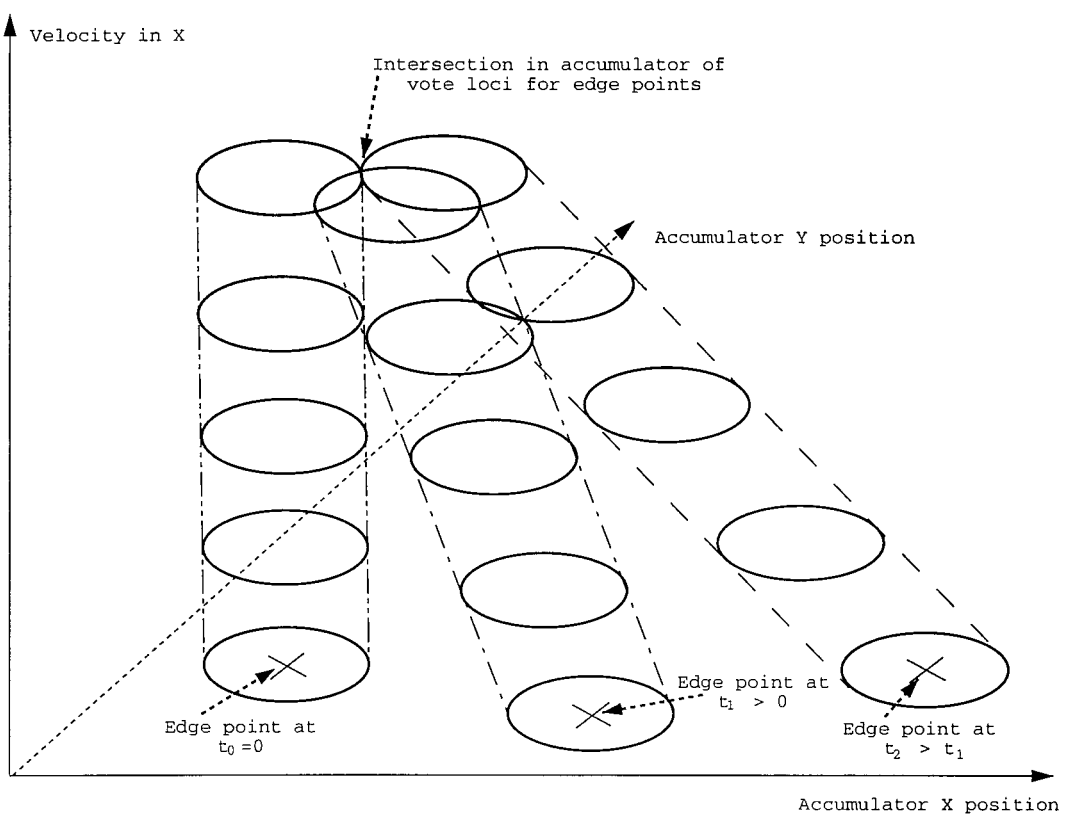


Figure 3: View of a VHT accumulator for a moving circle

Pseudocode for the VHT for circles algorithm (allowing linear x and y motion and a range of possible circle radii) is given below:

```
For (f = first frame -> last frame)
 For (all edge pixels e_x, e_y in frame f)
  For (r = r_min -> r_max)
   For (v_x = v_x_min -> v_x_max)
    For (v_y = v_y_min -> v_y_max)
     For (theta = 0 -> 359)
      a_x = e_x - v_x * time[f] - r * cos(theta)
      a_y = e_y - v_y * time[f] - r * sin(theta)
      accumulator[a_x, a_y, r, v_x, v_y] ++
```

2.4 Optimisation

No development involving the HT is complete without at least a brief examination of optimisation techniques. Since the process that guarantees optimality is a form of exhaustive search (albeit efficiently implemented), the computational resources required by the algorithm will always be greater than those used by less comprehensive algorithms. As a result of these requirements, there is a substantial body of work directed towards alleviating the computational burden. The approaches fall into three basic responses to the problem: architectural, pragmatic and analytic.

The first, architectural, is the simplest and depends on the availability of faster computers. This takes the form of parallel architectures or reliance on Moore's Law, the general trend of doubling of computing power about every eighteen months. The HT is known to be well suited to parallel implementation, with many options for splitting the processing (e.g. by image, by region of an image, by ranges of parameter values, etc). Leavers' survey [38] has a good summary of parallel implementations of the HT. With other application domains, the increase of available computing power has eventually caught up with demands, proving this solution to be an adequate one in some cases, albeit one that requires a measure of patience. In many respects, the ever-increasing speed of computers has made this research possible - only a few years ago, the computational requirements would have placed it beyond the reach of all but the most well equipped (or patient!) organisations.

The second approach, pragmatism, involves applying heuristics, speed-ups, and memory-reduction techniques that may undermine the underlying principle of exhaustive search and hence the robustness of the algorithm. For example, multi-stage processing, "pyramidal" methods (e.g. the Adaptive HT [29], Hierarchical HT [51,62]), random sampling algorithms such as the Randomised HT [60] and genetic algorithms (Section 5.2) fall into this category.

Probably the most common and effective method is to reduce the number of points to be considered. This can be done robustly and simply by discarding any points that definitely will not contribute to a meaningful analysis (e.g. highly isolated single pixels when searching for lines). Removing meaningless feature points has the twofold benefit of reducing accumulator noise due to false votes and reducing the

search space, resulting in a general speed improvement proportional to the number of points rejected.

One of the most prominent uses of robust point reduction is the GHT. The gradient direction of a point in the search image constrains the parts of the template shape drawn into the accumulator to just those with a matching gradient direction. Before the GHT, each feature point matched, and voted for, all points in a template.

Points can also be discarded in a non-robust way by, for example, using a probabilistic strategy that picks a random subset of feature points (e.g. the Randomised Hough Transform [60]). This method will still give dependable results provided that the sample size is large enough and random enough to gather sufficient evidence to identify features in the noise. The size of the feature under detection is a significant factor in the reliability of probabilistic strategies - as the ratio of feature size to noise decreases, random picking of points is more likely to miss the (relatively few) important feature points. One of the problems with this technique is deciding when enough data has been considered to give an accurate result (the stopping criterion). Balancing the speed of execution with accuracy can be difficult and is often heuristic.

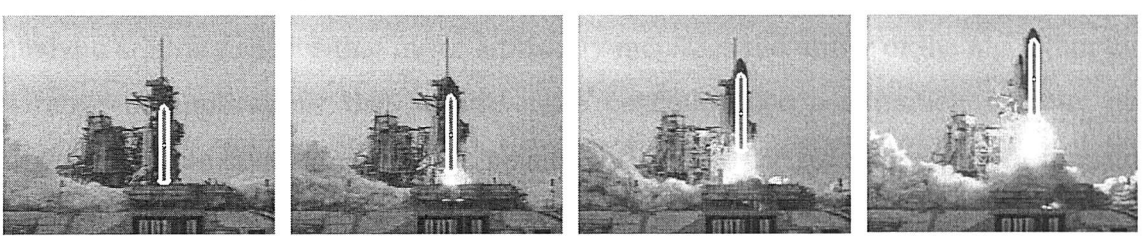
Other pragmatic improvements can be made purely in the implementation domain. For example, the space requirements of the accumulator can be reduced by using sparse arrays or linked lists, which only allocate space for array cells that are in use, and computation times can often be significantly reduced by caching important data structures, etc.

Finally, the analytic approach tries to reduce the computation requirements by using extra information to make the problem simpler (decomposing it), often shifting some of the work to pre- or post-processing stages (also common in pragmatic approaches). One approach to parameter space decomposition uses sets of points to constrain voting. For example, in [3], Aguado describes a method where two points are picked at random and a third point found along a line perpendicular to one of the endpoints of a baseline between the two points. These three points, along with some stored angular information, form a geometric invariant that can locate the correct vote point for the template shape. A disadvantage is that since multiple points are required per vote, the processing burden for each vote increases considerably. Furthermore, all three points must belong to the same shape or any votes cast are just noise. "Windowing" is a partial solution to this problem – points are selected from a local

region to increase the chance that they will all belong to the same primitive. Finally, as the complexity of the transformations applied increases, geometric invariants become more and more general, requiring more processing to produce. For example, an affine transform destroys the relationship between angles of a triangle and invalidates Aguado's method, which is limited to similarity-transformed shapes.

2.5 Contributions

We will describe a new technique for extracting moving arbitrary shapes, which has been created by fusing the two evidence-gathering techniques, the VHT and Fourier-descriptor template representation. Uniting these techniques unifies their unique and complementary advantages. The Fourier Descriptors provide a continuous template representation, minimising discretisation error in the algorithm, and the VHT component exploits the temporal correlation across a sequence, mitigating the effects of noise and occlusion. The new algorithm does not require initialisation or training and avoids the need to solve the correspondence problem, inheriting these characteristics from the VHT. For illustration, Figure 4 shows frames of a sequence where the location, velocity and acceleration parameters of a Space-Shuttle booster during launch were correctly extracted by the new technique. See Section 3.4.6 for a fuller breakdown of this extraction.



(a) Frame 0 (b) Frame 10 (c) Frame 18 (d) Frame 26

Figure 4: Space-Shuttle launch with extracted template (of the booster) superimposed in white

However, the CVHT (for convenience, we will refer to the continuous-template variant of the VHT as the CVHT) is still limited by its parameterised motion model. If the Shuttle imagery included a parabolic trajectory, this more complex motion would have to be incorporated into the motion model and would boost the dimensionality considerably. Hence, the simplistic approach to improving the generality of the

motion model is to increase the complexity of the HT kernel to represent an increasingly complex motion path. Consequently, an accurate polynomial description of an arbitrary path will require a large or even infinite number of terms, massively increasing the dimensionality of the problem. There are parallels to this parameterisation of motion in the earlier parameterisation of shape, where increasingly complex shapes were represented by more complex parameterisations and a commensurately larger dimensionality. The solution to this dimensional explosion was found in the use of templates, which allowed an efficient and low dimensional parameterisation of any shape. The cost of this approach is that the method loses the (debatable) flexibility of finding all descriptions of all possible shapes in a scene. Following this historical parallel, the remaining part of our new approach is to describe the motion by a template, like the shape itself. These "motion templates" extend the use of templates in the HT from the spatial domain into the temporal. This ameliorates the punitive computational burden associated with increasing dimensionality since the aim changes from finding the potentially unlimited set of parameters that characterise a particular motion to finding the limited set of parameters that locate the object undergoing the specified motion.

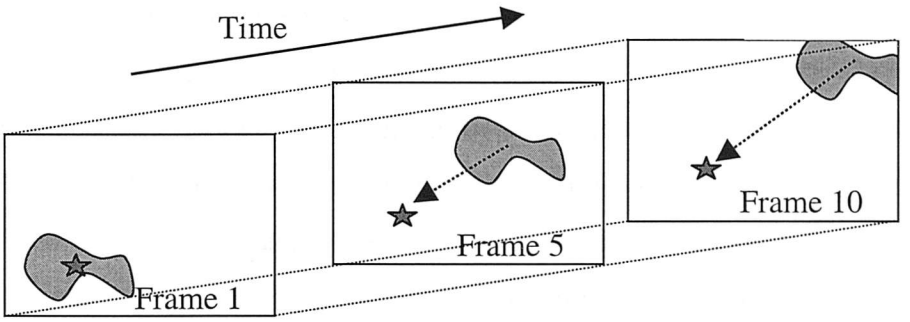
These novel developments clearly add significant functionality and flexibility to the VHT – removing the limitations of shape modelling by analytically described conic sections and parametric motion modelling. Adding the capability to extract non-analytic arbitrary shapes that move arbitrarily increases the utility of the algorithm to a range of applications that require more general shape and motion models. In particular, these developments have opened the door to the use of the HT for human gait analysis.

3 Continuous VHT

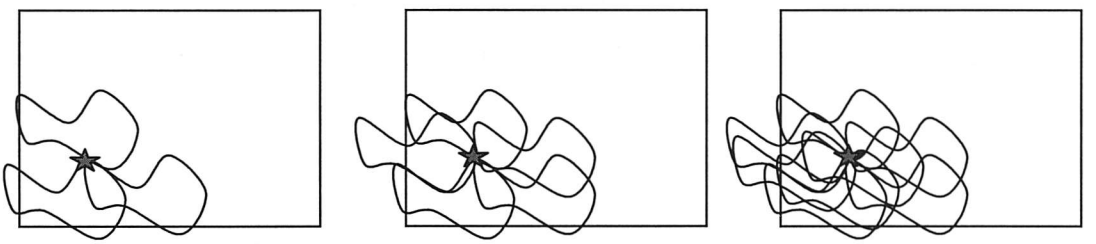
In this chapter, we will illustrate the implementation of the continuous VHT (CVHT) with some visualisations, present some theory describing the implementation and discuss results showing its performance in varying circumstances. We then test it in comparative evaluation with GHT-based techniques.

3.1.1 Implementation

In order to add arbitrary shape extraction to the VHT, the Fourier descriptor version of the GHT is extended in the same way as the HT for circles was extended into the original VHT - by introducing velocity terms to the shape description. Instead of drawing a motion-compensated circle in the accumulator (as in the VHT), the Fourier descriptors are used to trace a locus of votes in the form of the template shape, adjusted for the estimated motion of the object. The accumulation process for the sequence in Figure 5a is illustrated in Figure 5b-g, with votes increasing as more frames are added. Here we use Merlin and Farber's [39] variant of the voting process – voting for all the points in the template rather than a restricted set such as in the GHT. Figure 5b-d shows the plane of the accumulator for the correct estimate of velocity while Figure 5e-g shows a plane with an incorrect velocity ($x = 0$). Reflected instances of the template shape are generated in the accumulator, centred on motion-compensated edge pixel co-ordinates from the frame being processed. Shapes drawn in the accumulator planes are reflected because co-ordinates describing the template are subtracted from the co-ordinate of the edge pixel. This is because we are reversing the vector from the reference point to the edge point so that we can locate the reference point from the edge pixel. The net effect is that a template shape is reflected around both axes when drawn in the accumulator planes.



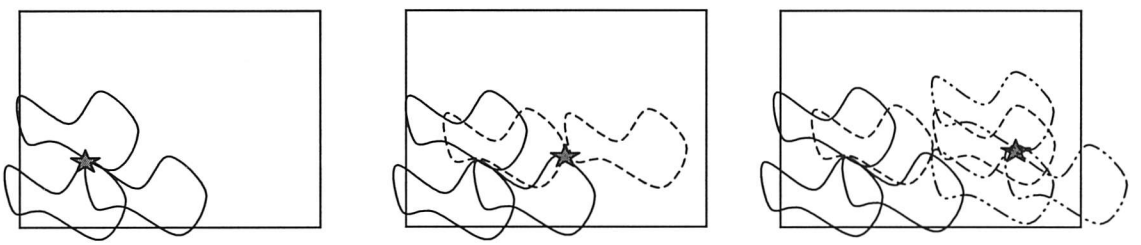
(a) Simulated sequence of linearly moving arbitrary shape (star = start point)



(b) Accumulator plane for correct velocity after processing frame 1

(c) Accumulator plane for correct velocity after processing frame 5

(d) Accumulator plane for correct velocity after processing frame 10



(e) Accumulator plane for incorrect x velocity after processing frame 1

(f) Accumulator plane for incorrect x velocity after processing frame 5

(g) Accumulator plane for incorrect x velocity after processing frame 10

Figure 5: Simulated sequence and accumulator planes deriving from it

The motion compensation mentioned above is simply back-projection along the expected line of motion, thus converting the co-ordinates to the same temporal frame of reference as the initial frame (at frame time $t = 0$). Hence, votes for a particular edge pixel in frame 5 fall in the same place as those for frame 1. Once the voting process is complete, intersections of template shapes in the accumulator form peaks

that indicate the location (at frame time $t = 0$) of an instance of the shape in the sequence. The more coincidences that occur (whether due to correct velocity estimates or chance), the stronger the resulting peak.

The dimensionality of the algorithm can be considered in two sections. The shape section of the CVHT requires four parameters - x and y offsets, scale and rotation - allowing it to cope with shapes that have undergone a similarity transform. The number of parameters required for the motion section varies with the complexity of the model. The motion model used by the CVHT may be changed by altering the kernel of the HT (see equation 6 for a linear velocity kernel). Here, we have generally used one of the simplest models - linear velocity in one or two axes - in order to limit the dimensionality of the problem. More complex motion models may require more parameters to be searched for. For example, to accurately extract an object that has linear velocity and acceleration requires two parameters per axis (velocity and acceleration). As the number of parameters rises, dimensionality increases proportionally. In terms of visualising the extraction process (as in Figure 5), changing the motion model alters the motion compensation stage to perform a more complex back-projection of the anticipated motion (e.g. from linear back-projection to linear with acceleration back-projection).

3.1.2 Computational cost

The computational complexity of the CVHT is dependent on the motion model chosen. The basic linear velocity variant is of order $O(N^6)$ in terms of algorithmic complexity or, in terms of the number of operations, $O(\#points \times \#s \times \#r \times \#v_x \times \#v_y)$, where *#points* is the number of feature points (e.g. thresholded edge-pixels in a 2D image) in the sequence, $\#s$ and $\#r$ are the number of discrete steps in the parameter ranges for initial shape scale and rotation and $\#v_x$ and $\#v_y$ are the number of discrete steps in the parameter ranges for velocity in the x- and y-axes respectively. It may be more useful to consider the algorithmic complexity as consisting of two parts: the shape description part – $O(N^4)$ - and the motion description part - $O(N^2)$ for a linear velocity kernel. Combining these gives the previous result $O(N^4 \times N^2) = O(N^6)$. Note that the CVHT has fixed costs for the shape description part due to the template-based modelling – only the motion description has variable cost. In other words, any shape (including affine transformations) can be represented in the same size of accumulator (i.e. in four dimensions), but different forms of motion require different numbers of

accumulator dimensions (e.g. two dimensions for linear x and y velocity, two more dimensions to include x and y acceleration, etc).

The CVHT has the normal penalties of being derived from the HT - namely high memory and computational requirements. In mitigation, the optimisation procedures described in Section 2.4 are applicable and can bring the cost to a reasonable level. The crucial difficulty with the CVHT is that the dimensionality of the problem increases with the complexity of the motion model, a problem we address in Chapter 4 with motion templates.

3.2 Theory

The theory presented in this section is developed from the Fourier-descriptor variant of the GHT [4] discussed earlier.

In order to develop the voting mechanism, we require an arbitrary-curve parameterisation for shapes. As stated previously, we have chosen elliptic Fourier descriptors for this purpose (see Section 2.2.1 for some relevant theory). The curve shall be represented by the vectors $\upsilon_x(s,\overline{FD}_x)$ and $\upsilon_y(s,\overline{FD}_y)$ (as defined in equation 2, but omitting the DC bias as discussed in Section 2.2.1). The shape template's scale and rotation is given by $\mathbf{a}_s=[l_g\ \rho_g]$, so the scaled and rotated shape can be described as

$$R_x(s,\mathbf{a}_s)=l_g\upsilon_x(s,\overline{FD}_x)\cos(\rho_g)-l_g\upsilon_y(s,\overline{FD}_y)\sin(\rho_g) \tag{5}$$

with a similar equation for R_y. Now, we require a kernel that defines the shape of votes to be cast in the accumulator for each feature point (e.g. an edge pixel). This is a combination of curves, each with its origin on the reference point to be voted for (typically at the centre of the template shape) but offset by the velocity, and at a number of orientations and scales (for similarity transform invariance). This combination of curves can be obtained from:

$$\overline{\omega}(s,T_{ref},l,\rho,v_x,v_y)=R_x(s,l,\rho)U_x+R_y(s,l,\rho)U_y+T_{ref}v_xU_x+T_{ref}v_yU_y \tag{6}$$

where v_x and v_y are respectively the x centre and y centre velocity parameters, U_x and U_y are two orthogonal vectors defining the x- and y- axis respectively and T_{ref} is a time reference relative to the arbitrary start point (e.g. frame number versus frame 0). This curve is inserted into the accumulator by offsetting it from the co-ordinates of each feature point in the image sequence IS, defined by:

$$IS=\left\{\overline{\lambda}(P,T_{ref})\,|\,P\in D_P,T_{ref}\in D_T\right\} \tag{7}$$

Here, $\bar{\lambda}(P,T_{ref})$ is a function that defines the points in the image sequence for a frame time T_{ref}, where a suffix on the domain indicates its extent (here, D_P is the domain of the pixels in an image in the sequence and D_T is the domain of the frame times (i.e. number) in the sequence). The accumulator vote-pattern expression is then:

$$A_{P,T_{ref}} = \left\{\bar{\lambda}(P,T_{ref}) - \bar{\omega}(s,T_{ref},l,\rho,v_x,v_y) \mid s \in D_s\right\}; P \in D_P, T_{ref} \in D_T \quad (8)$$

These equations describe the concept of the HT but do not formalise the actual technique used, namely the accumulation phase. The parameter space can be mapped into an accumulator by using a matching function, which determines whether a point, $\bar{c}$, in parameter space should be incremented for a point, $\bar{d}$, in the set $A_{P,T_{ref}}$. The equation below defines the simplest accumulation strategy, namely incrementing an accumulator cell by unity for each match. Changing the matching function M can accommodate more complex strategies (e.g. the Fuzzy HT's more complicated voting strategy that casts multiple points in parameter space for each parameter set). In this thesis, we have exclusively used the simple case.

$$M(\bar{c},\bar{d}) = \begin{cases} 1 & \text{if } \bar{c} = \bar{d} \\ 0 & \text{if } \bar{c} \neq \bar{d} \end{cases} \quad (9)$$

This function simply maps points from the set $A_{P,T_{ref}}$ (i.e. the vote-pattern, offset from a particular feature point) to their exactly corresponding co-ordinates in parameter space. Next, this function is applied to $A_{P,T_{ref}}$ for a range of parameter values. This defines the continuous form of the CVHT, accumulating evidence in a parameter space S_{CVHT} according to:

$$S_{CVHT}(\bar{b},l,\rho,v_x,v_y) = \iiint M(\bar{b}, \bar{\lambda}(P,T_{ref}) - \bar{\omega}(s,T_{ref},l,\rho,v_x,v_y))ds\,dP\,dT \quad (10)$$

where $\bar{b}$ is the translation vector (i.e. the location of the reference point). Finally, this parameter space is sampled into a discrete multidimensional array S_{DCVHT}, which is expressed by:

$$S_{DCVHT}(\bar{b},l,\rho,v_x,v_y) = \sum_{T_{ref}\in D_T}\sum_{P\in D_P}\sum_{s\in D_s} M(\bar{b}, \bar{\lambda}(P,T_{ref}) - \bar{\omega}(s,T_{ref},l,\rho,v_x,v_y)) \quad (11)$$

This expression gives an accumulation strategy for finding moving arbitrary shapes. For each edge pixel in a frame, a locus of votes is calculated from the Fourier description of the template shape and entered into the unified accumulator. The co-

ordinates of the loci are adjusted to allow for the predicted motion of the shape, dependent on the frame index, as in the VHT.

3.3 Pseudo-code algorithm

The algorithm that describes the CVHT voting process is:

```
For (frame f of the sequence)
 For (edge pixels ex, ey)
  For (scale=scale_min -> scale_max)
   For (angle=angle_min -> angle_max)
    For (t=0 -> 2*PI)
      Generate vector (ax, ay) from FD (t)
      Rotate and scale (ax, ay) vector
      For (vx=vx_min -> vx_max)
       For (vy=vy_min -> vy_max)
        Offset (ax, ay) vector from (ex, ey)
        Calculate velocity offset from f, vx and vy
        Subtract offset from (ax, ay) vector
        accum[ax, ay, scale, angle, vx, vy] ++
```

3.4 Results

3.4.1 A short note on pre-processing

While a considerable proportion of the test sequences used in this thesis are synthetic (in order to highlight the particular analysis that is being pursued), some are from real-world cameras and have been pre-processed. Typically they are edge-detected using the Canny operator [11], with an appropriate set of parameters for the sequence in question (i.e. judged by eye to give moderately clean edges around the central object of interest without excessive noise from small intensity gradients). Normally, these parameters tend to be as follows: low hysteresis threshold = 0.4 (maximum intensity is 1.0), high threshold = 0.6, standard deviation of the Gaussian kernel = 1.5.

An additional pre-processing step used below is the addition of artificial noise. Whilst most noise models are explained in the text appropriate to the context, the addition of Gaussian image noise may cause confusion because two forms are used. In this thesis, image noise is typically generated by adding a Gaussian-distributed value to individual pixels, often with a uniform random variable defining how much of the image is affected (in the results below, this uniform random variable is normally reported as the quantity of noise added, e.g. "40%", meaning approximately 40% of the pixels will have been corrupted). However, the resulting pixel value may be beyond the acceptable range. In this case, we have two models: clipped and wrap-around.

"Clipped" Gaussian noise means that out-of-range pixel values are clipped to the nearest limit, so a pixel value of 1.4 (whiter than white) would be clipped to 1.0 (full white) and a value of –0.3 would be clipped to 0.0 (full black). When this type of noise is applied to binary images (such as edge images), there is a better than 50% chance that a pixel will be unchanged because of the combination of this clipping operation and the binary nature of the image. If the original pixel is full white and more white is added (50% chance due to the mean of the Gaussian being 0), then the clipping will reduce the pixel value back to full white (i.e. unchanged). If "black" noise is added to a full white pixel, then the amount added must be greater than the binarisation threshold required to flip the pixel to full black - typically a halfway

threshold at 0.5 (mid-grey). A similar problem occurs if the original pixel is full black.

"Wrap-around" Gaussian noise partially solves this problem by joining the pixel-value range at both extremes. So, an out-of-range pixel with a value of 1.4 becomes a pixel with a value of 0.0. The consequence of this noise model is that pixel values are unlikely to remain the same in over 50% of cases. In fact, there is a slight tendency for pixel values to become inverted. Degradation of edge-based algorithms, such as the ones presented in this thesis, is far worse with this noise model because, at 100% noise, there is a high probability that the majority of the original edges will have been inverted. Effectively, useful edges tend to be removed whilst false ones appear.

The reason for these two models is that, during the original work on the CVHT, the "clipped" model was found to provide sufficient noise levels for testing. Later work with motion templates improves on the CVHT to the extent that the "clipped" model's 100% noise was insufficient to cause extraction errors, hence the requirement for more powerful noise. While this is to some extent an artificial situation, it is true to say that many noise models do not bear much resemblance to real-world noise. In this thesis, we were mainly interested in finding the limits of the techniques tested, and used noise models appropriate to this requirement.

3.4.2 Simulated image noise

The CVHT was run on a small five-frame sequence (0% noise line in Figure 6 below) showing a shape moving linearly along the x-axis at a velocity of one pixel per frame. The low resolution (20×20) was chosen to make practical computation of large-scale tests. Noise was added at random to each frame of the sequence at eleven noise levels from 0% to 100% random coverage of the frame, in steps of 10%. The noise distribution was zero-mean Gaussian with a standard deviation of three and used clipping when a pixel value exceeded the maximum allowable (i.e. "clipped" Gaussian noise was used). Examples of the effects of the increasing noise levels can be seen in Figure 6. The grey-level images produced by the application of noise are shown in the original state before being thresholded by the algorithm. For comparison to a standard technique, we have generated results for the normal GHT tracking algorithm (i.e. a standard GHT is applied to each frame of the sequence separately and the results are combined using linear regression to calculate the velocity terms). The test conditions were as described above.

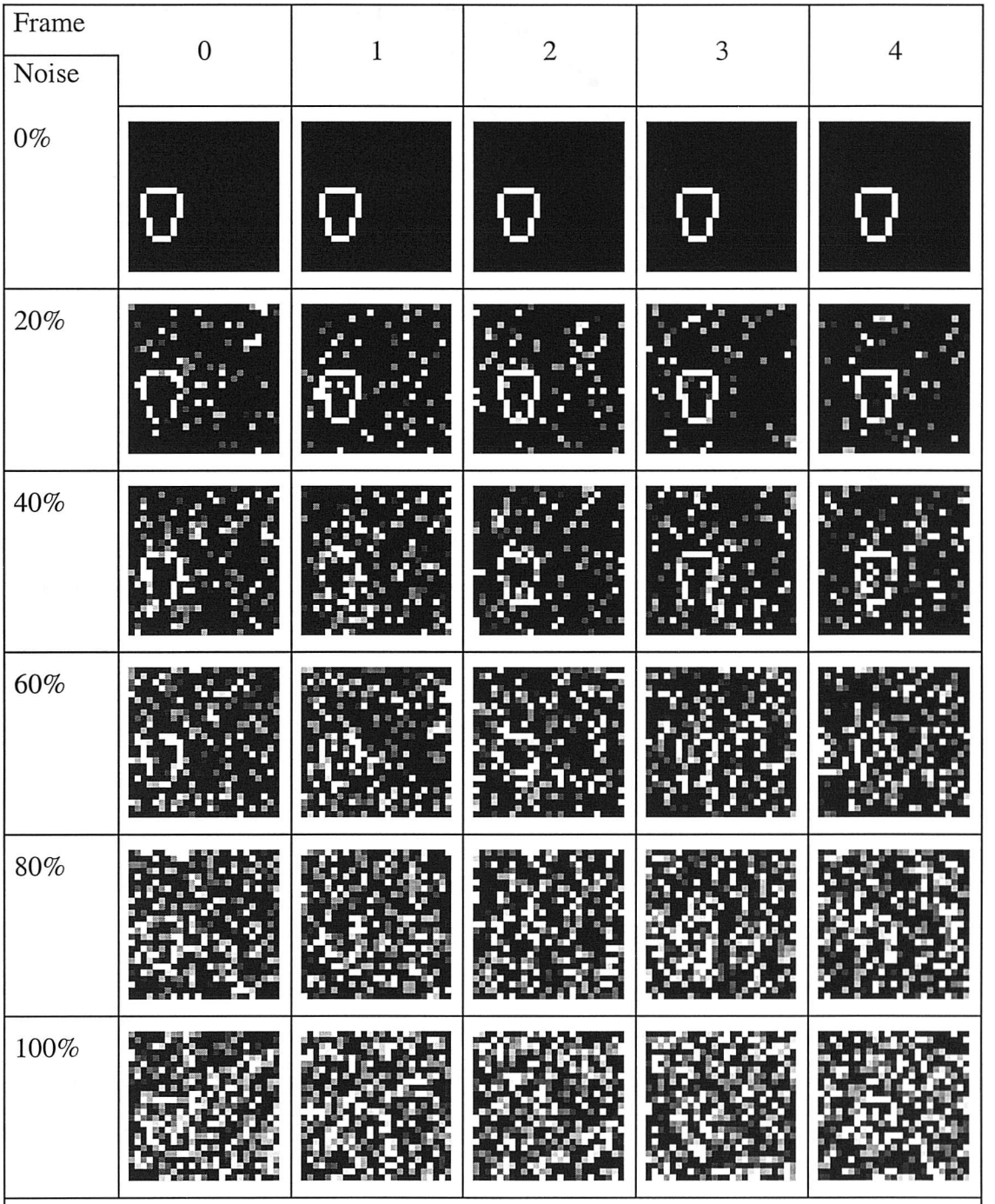


Figure 6: Artificial sequence with added Gaussian noise

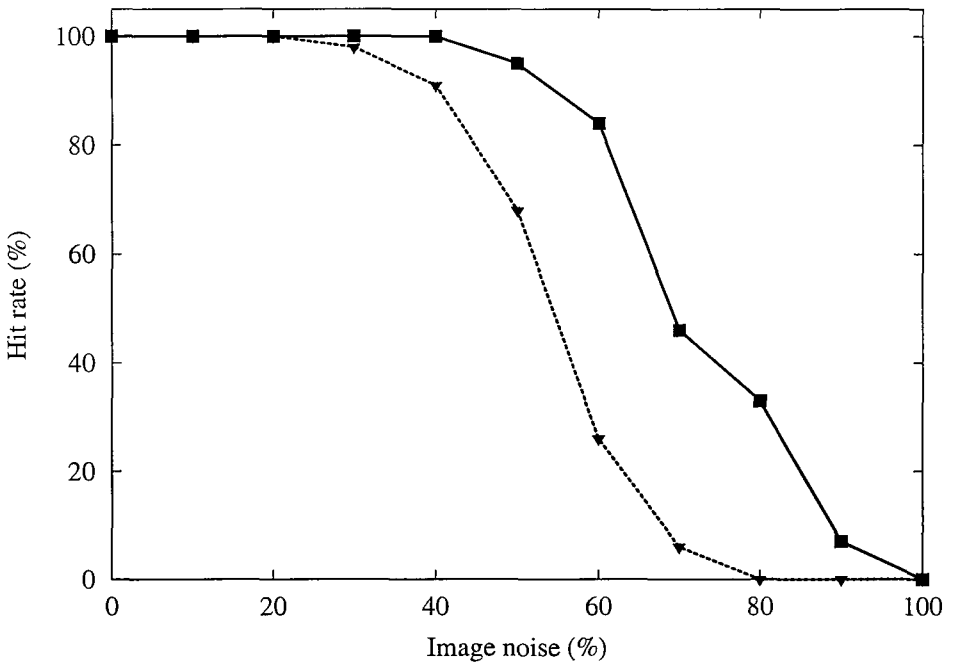


Figure 7: Noise performance (dashed line with triangles = GHT-based, solid line with squares= CVHT)

The graph (Figure 7) shows that the CVHT is significantly more accurate (in terms of exact extraction accuracy) than the GHT-based technique. The vertical axis on the graph is hit rate, or the percentage of correct extractions for multiple (100 in this case) trials with different random noise at each image-noise level. Whilst the GHT-based technique starts to lose accuracy at 30% noise, 15% more noise can be added before the CVHT begins to lose accuracy. Although both techniques fail in extreme noise as expected, the CVHT is still reasonably successful at 80% noise whereas the GHT-based technique fails completely. It is interesting to note that while the shape in the static pictures in Figure 6 appears hard to perceive to human vision at 60% noise, when it is presented in an animated form it is actually perceivable at noise levels of 80%. This effect is exploited by the new technique when it accumulates temporally correlated evidence, enabling it to handle noise levels that are approximately twenty percent greater than the GHT comparator. The GHT based technique is limited to the amount of evidence available in a single frame. When the noise becomes sufficient to mask out the correct peak in a single frame, the GHT technique is left with effectively random results to process for velocity terms. In intermediate noise levels, some of the results for each frame may be incorrect and this will tend to throw off the final regression step. The integrated approach taken by the new algorithm is more global and is not so susceptible to corrupt frames as is demonstrated strongly in the occlusion

testing later. This is consistent with earlier observations comparing the VHT to the HT with linear regression [43].

3.4.3 Simulated occlusion

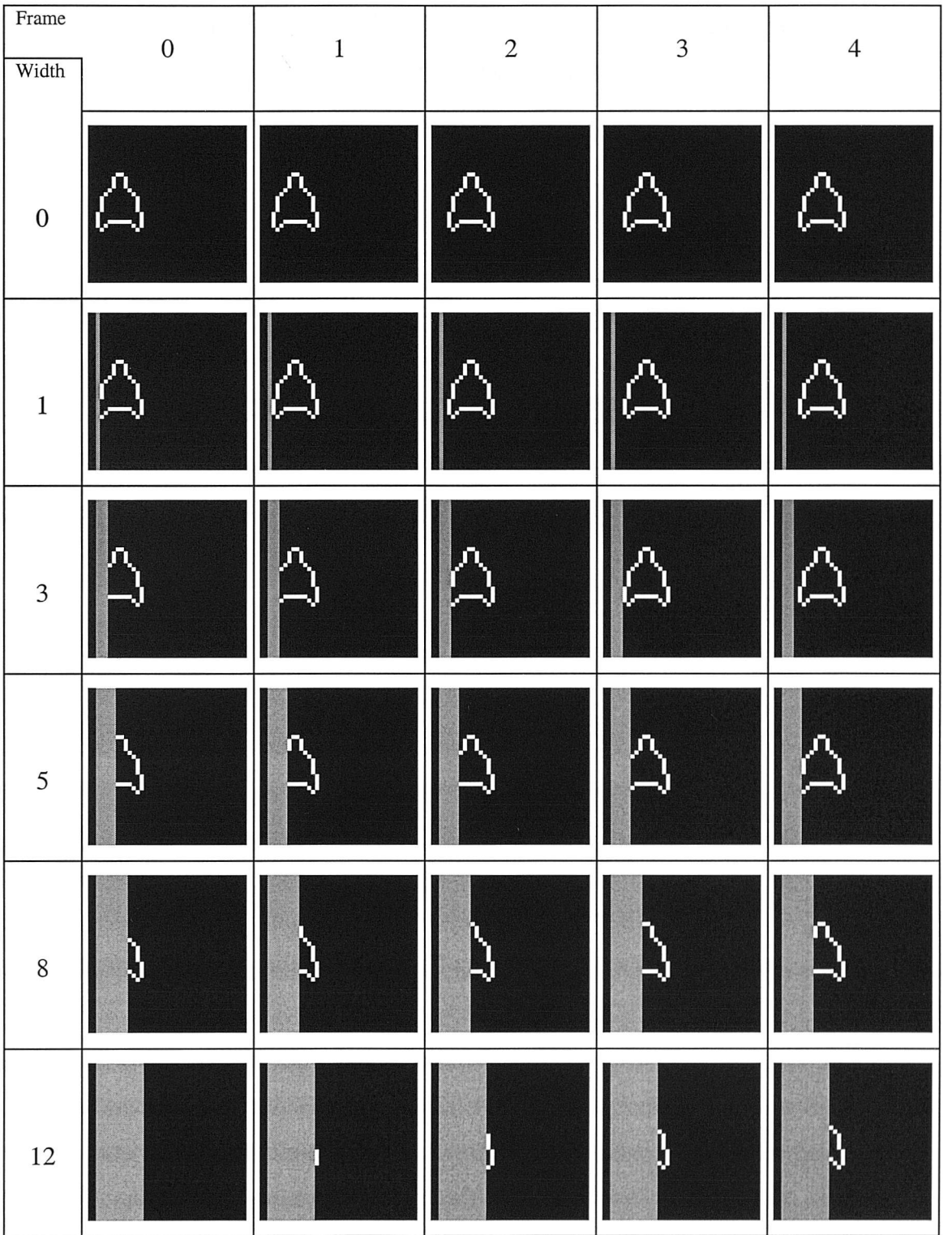


Figure 8: A sample sequence showing several levels of occlusion

A simple test of the effects of occlusion was carried out on the five-frame sequence shown above. No image noise was added to the sequence since this is purely a test of simulated occlusion. Instead, a varying number of vertical lines of pixels starting from column two were blanked out. In effect, the generated sequences will show the target shape emerging from increasingly serious occlusion. With the HT, it does not make any difference whether the shape starts occluded or passes into occlusion – the net effect is the same; a certain percentage of the sequence is obscured. Figure 8 shows example frames from the occluded sequence with varying widths of occlusion bar (the bar is shown in grey to make it visible although normally it would be black to blank out the occluded columns). Again, we have used the standard GHT tracking algorithm as a comparator.

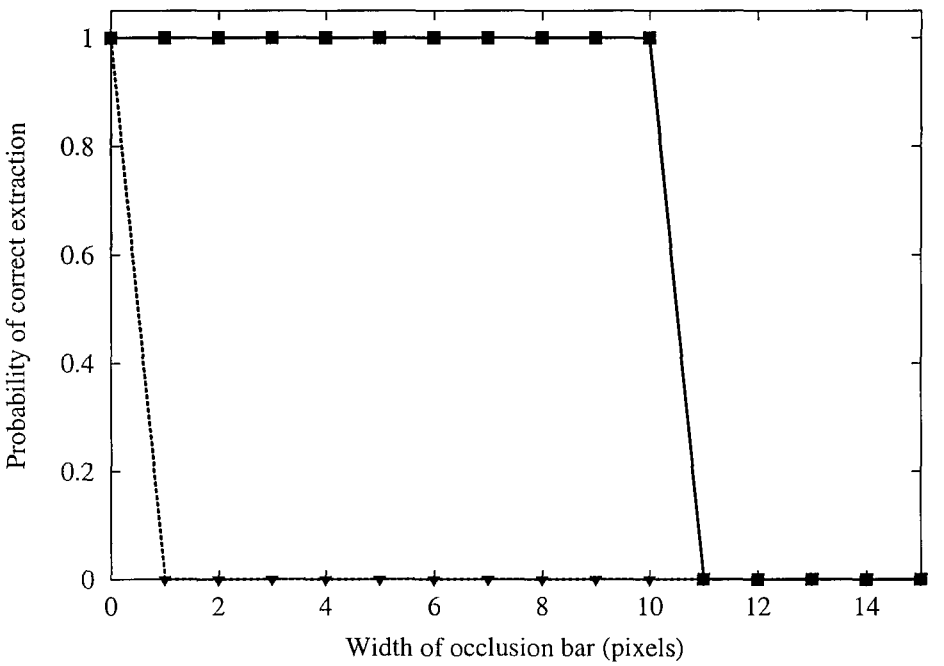


Figure 9: Occlusion Tests (dashed line with triangles = GHT-based, solid line with squares = CVHT)

The results shown in Figure 9 reveal that the new technique keeps track of the shape until the blanking is eleven pixels wide – almost obscuring the shape entirely at the start of the sequence, and showing only a few pixels of it for the remainder of the sequence. Surviving this level of occlusion is an excellent result but unsurprising since, due to the synthetic nature of the sequence, the target is surrounded by empty

space. Therefore the only competing noise in the accumulator comes from the shape itself.

The GHT based algorithm failed as soon as any blanking was introduced. This failure reveals more about the algorithm's implementation than about its resilience to occlusion. The current implementation uses the estimated location of the template shape in every frame as an input to the linear regression stage. Therefore, when a frame is corrupted and gives an incorrect result, the output of the linear regression stage is affected causing a global estimation error. A more sophisticated implementation might include a heuristic that ignores frames giving evidence inconsistent with the majority of frames.

The earlier results [43] relating to the VHT should also be applicable to the new technique since the underlying characteristics are essentially unchanged. These results indicate that VHT derived algorithms are capable of handling even extreme occlusion due to the global integration of evidence across the entire sequence.

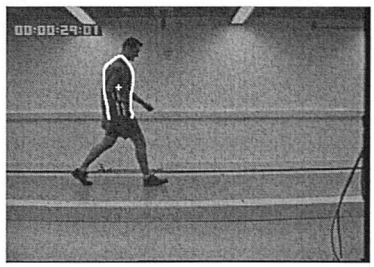
3.4.4 Real-world imagery: finding people

We now apply the CVHT to locate a moving human body in a sequence of images. The current implementation of the new technique locates rigid shapes moving with linear velocity. Clearly, its formulation is general so shape deformation could be included, as it was for pulsating arteries in the original VHT formulation [43], but this would be considerably more complex. In the case of a human walking, the torso is approximately a constant shape and, if the camera is far enough away, the bobbing motion of gait is small enough to be compensated for by the resilience of the evidence gathering approach. Consequently, it is possible to detect people using the technique in its current form by searching for the torso. However, no meaningful gait data can be gathered from just the location of the torso so this method of locating a human silhouette is only useful as a primer for later (gait) analysis. Nonetheless, using the CVHT to locate a human demonstrates empirically that it is applicable to real world images.

Self-occlusion of the body due to the motion of the arms and legs is a problem that affects the performance of many person-tracking algorithms. By the nature of evidence gathering, the new algorithm copes with occlusions that do not reduce the correct peak below the level of noise in the accumulator. As a result, there is no immediate need for special precautions.

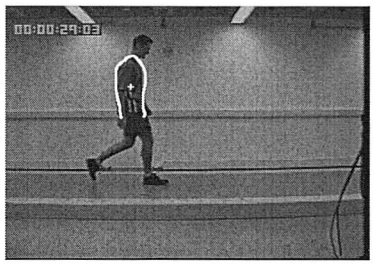
(a) Frame 1

(b) Frame 2

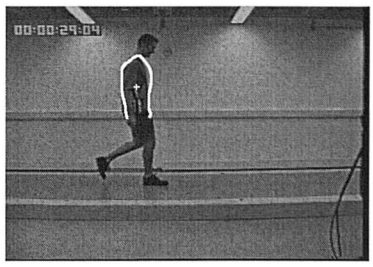


(c) Frame 3

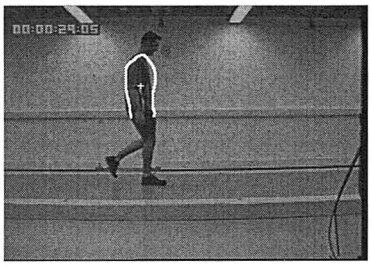
(d) Frame 4



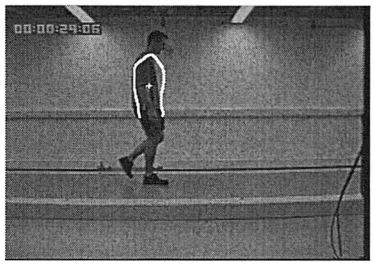
(e) Frame 5



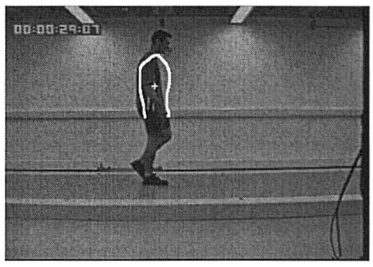
(f) Frame 6



(g) Frame 7

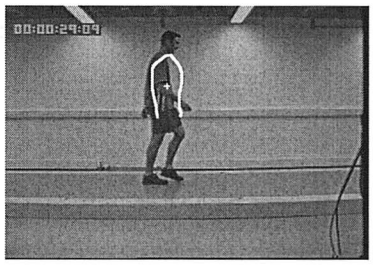


(h) Frame 8

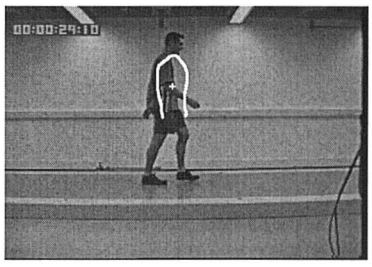


(i) Frame 9

(j) Frame 10



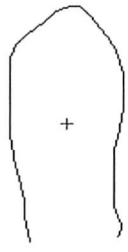
(k) Frame 11



(l) Frame 12

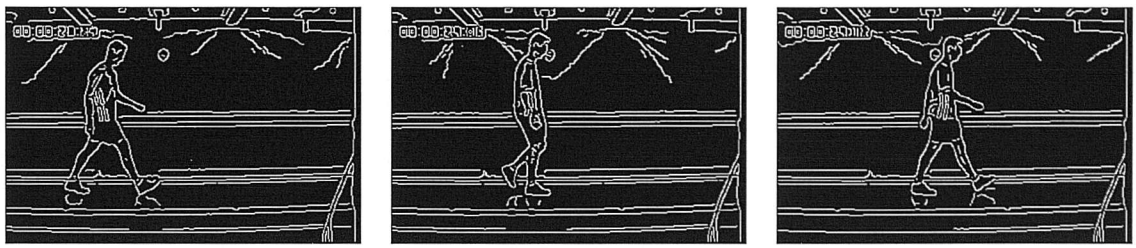
(m) Frame 13

(n) Frame 14



(o) Shape template

Figure 10: Extraction of MP1 sequence using a linear velocity model



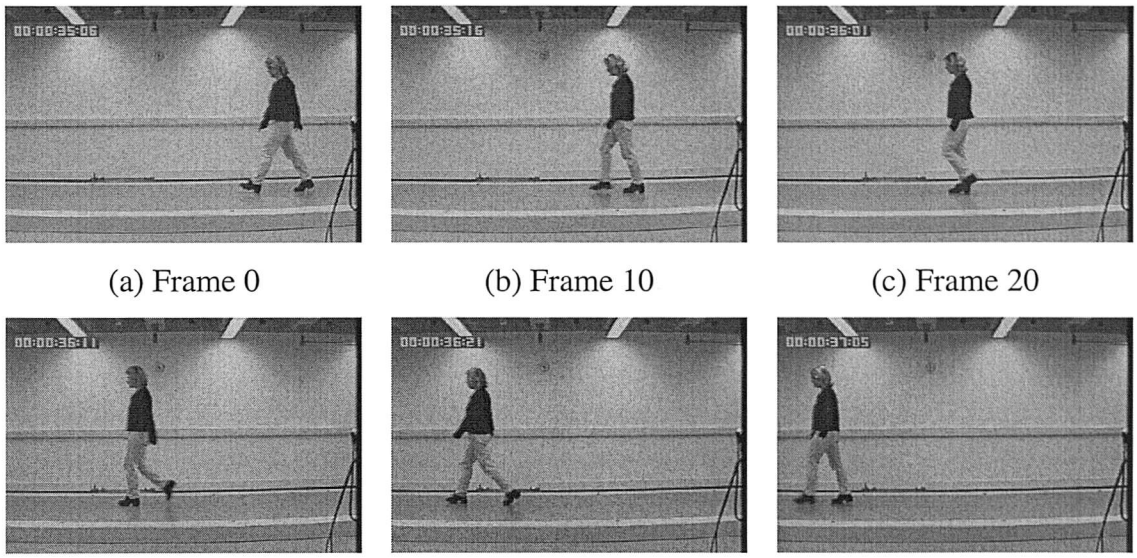
(p) Edge-detected frame 1 (q) Edge-detected frame 8 (r) Edge-detected frame 14

Figure 11: Sample edge-detected frames from the MP1 sequence

Figure 10o shows a reconstructed template of a walker's torso, which was originally created by manually tracing the torso in the first frame of the sequence. Also shown in Figure 10a-n are several frames of the MP1 walker sequence with the template superimposed on the extracted locations. Processing was actually performed on edge-detected instances of the images (some examples in Figure 11p-r), but the original images are shown here for clarity.

During the first part of the sequence, the walker's location is accurately extracted - the initial location is exact and the extracted speed (thirteen pixels per second) is correct. Shortly after frame seven the walker rises up on his leg (vertically, there is a rise of fifteen pixels), which will cause the votes to "miss" in the accumulator, since this movement has not been accommodated in the evidence gathering strategy. He also slows down slightly and this breaks the assumption of linear velocity and consequently the template "overtakes" the walker - frame 10 shows it some pixels ahead.

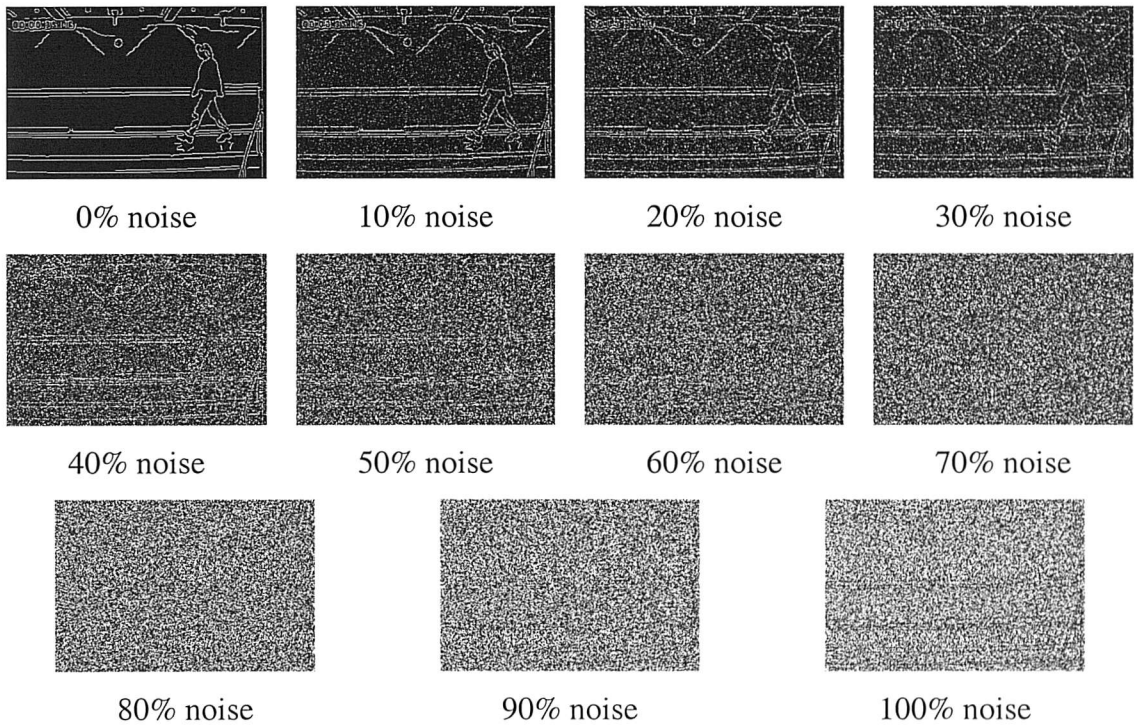
If noise is added, as in Section 3.4.1, or occlusion, as in Section 3.4.3, the performance drops off rapidly due to the large number of missed votes arising from the lack of modelling of the bobbing or vertical motion of gait. Under simulated addition of noise, as in Section 3.4.1 (Figure 7), the decline in accuracy is similar to that of the synthetic sequences. However, accuracy drops away at a lower noise level since the level of background noise is higher due to the surrounding scenery. Figure 14 shows the performance curve for a short range of tests (the curve is not smooth because practical concerns limit the number of trials that can be run) on a sequence of another walker (CA1). The CA1 sequence is particularly suitable for this analysis as the subject does not "bob" much. See Figure 12 for some typical frames from the CA1 sequence, and see Figure 13 for the result of various levels of (Gaussian wraparound) noise applied to the edge-detected first frame.



(a) Frame 0 (b) Frame 10 (c) Frame 20

(d) Frame 40 (e) Frame 50 (f) Frame 49

Figure 12: Frames from the CA1 sequence



0% noise 10% noise 20% noise 30% noise

40% noise 50% noise 60% noise 70% noise

80% noise 90% noise 100% noise

Figure 13: Frame 0 of edge-detected CA1 sequence with varying levels of wraparound Gaussian image noise

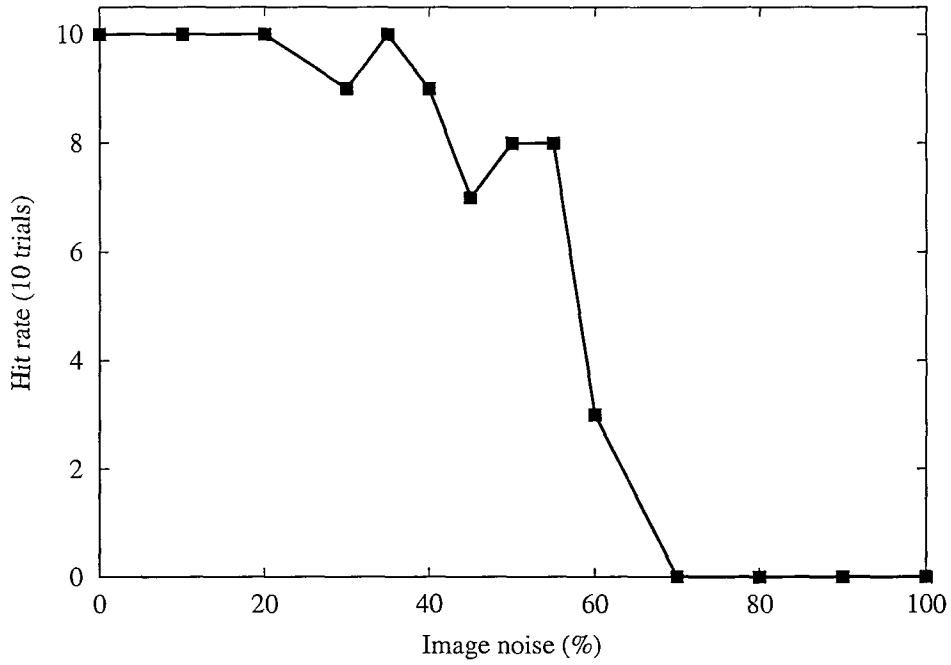


Figure 14: Gaussian noise tests on CA1 sequence

When the earlier comparison between GHT-based and CVHT extraction (Figure 7) is contrasted with Figure 14, it is clear that the former shows better performance than the latter. The threshold for good performance on simulated imagery was around 70% noise, whereas it has now dropped below 60%. This is simply because the results above are derived from a real-world sequence whereas the earlier results came from processing a synthetic sequence. In real-world imagery, non-target features increase the background noise in the accumulator and thus reduce the maximum noise tolerance before failure. With synthetic imagery, this factor can be excluded by removing any non-target objects.

3.4.5 Simulated time-lapse imagery

Here we have tested the performance of the CVHT in conditions that simulate time-lapse (i.e. infrequently sampled) video, as commonly used to store long periods of recorded footage. Time-lapse video can be viewed as regular occlusion of the target and, as such, will cause severe problems for techniques that suffer in occlusion. The CVHT has shown itself to be robust in occluded circumstances (Section 3.4.3) and demonstrates this again in these tests.

The first 20 frames of the CA1 sequence were used as the test sequence (see Figure 14 for the image noise analysis of this sequence). Time-lapse sequences were

simulated by discarding frames from the complete sequence. This is the same as keeping one of every *N* frames or only taking one frame for each time period that *N* full-rate frames would have occupied. Table 1 shows the relation between the number of frames kept and the percentage missing, which can be considered a measure of the degree of occlusion. Note that the percentage is variable - the "tends to" percentages are correct only if the number of frames in a sequence is an exact multiple of whatever *N* is required; e.g. a sequence with 3,628,800 frames (approx 40 hours at 25 frames per second) will be exact for all the values of *N* in Table 1. So for a seven-frame sequence, keeping one frame from every seven will give 85.71% missing (one kept and six of seven discarded). In the worst case, an eight-frame sequence, keeping one frame from every seven gives 75% missing (two kept and six of eight discarded). However, as the number of frames in a sequence increases, the percentages will approximate more and more accurately the numbers given below.

Number of frames kept (*N*)	**Percentage of frames missing (tends to)**	**Percentage for CA1 sequence (using 20 frames)**	**Percentage for CA1 sequence (using 40 frames)**
1 in 1	0%	0%	0%
1 in 2	50%	50%	50%
1 in 3	66.7%	65%	66%
1 in 4	75%	75%	74%
1 in 5	80%	80%	80%
1 in 6	83.3%	80%	82%
1 in 7	85.7%	85%	84%
1 in 8	87.5%	85%	86%
1 in 9	88.9%	85%	88%
1 in 10	90 %	90%	90%

Table 1: Relation between frames discarded and the percentage of occlusion

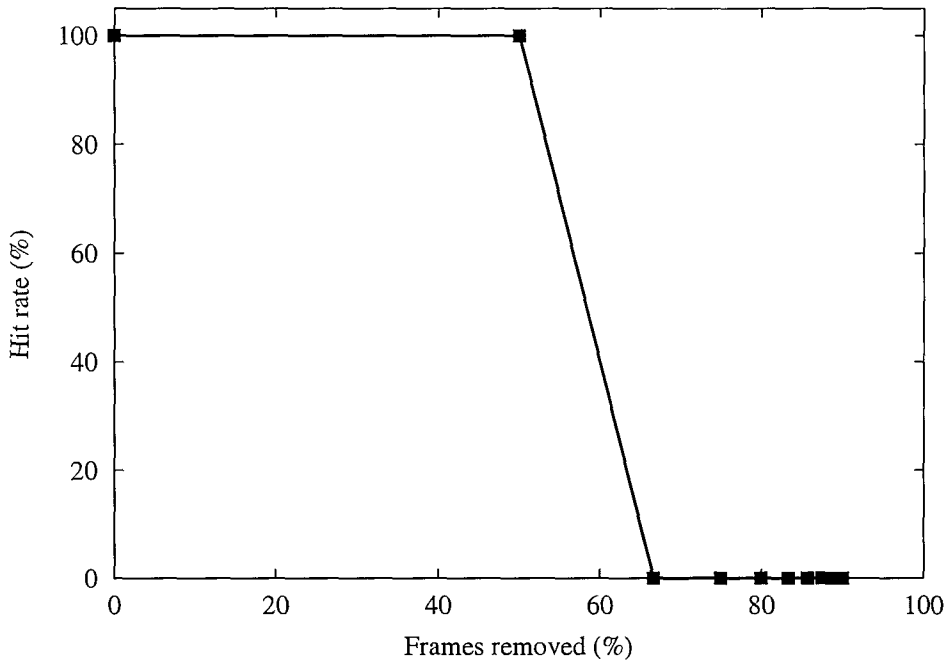
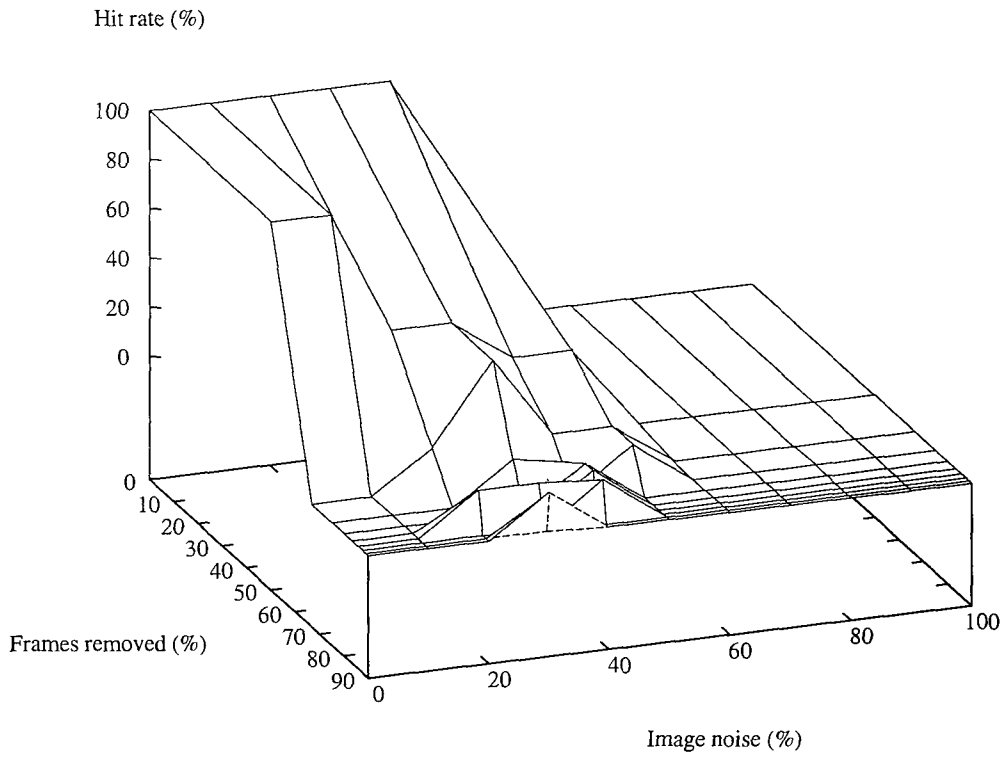


Figure 15: Simulated time-lapse for 20 frames of CA1 sequence

The performance curve in Figure 15 show that time-lapse does not have a significant effect until between 50 and 70% of the sequence is occluded. This equates to a failure occurring between losing half and two thirds of the frames. In addition to this test, we have examined the performance of the CVHT with noisy time-lapse video (i.e. time-lapsed noisy video).

As before, image noise is additive Gaussian noise (of the wrap-around form) with a mean of zero and a standard deviation of one applied to a uniform random percentage of the pixels. The number of trials is necessarily low due to time and computation constraints, resulting in unsmooth graphs. However, the number of trials varies across the spread of results with the majority of effort concentrated in the transition zones between guaranteed 100% misses and guaranteed 100% hits.



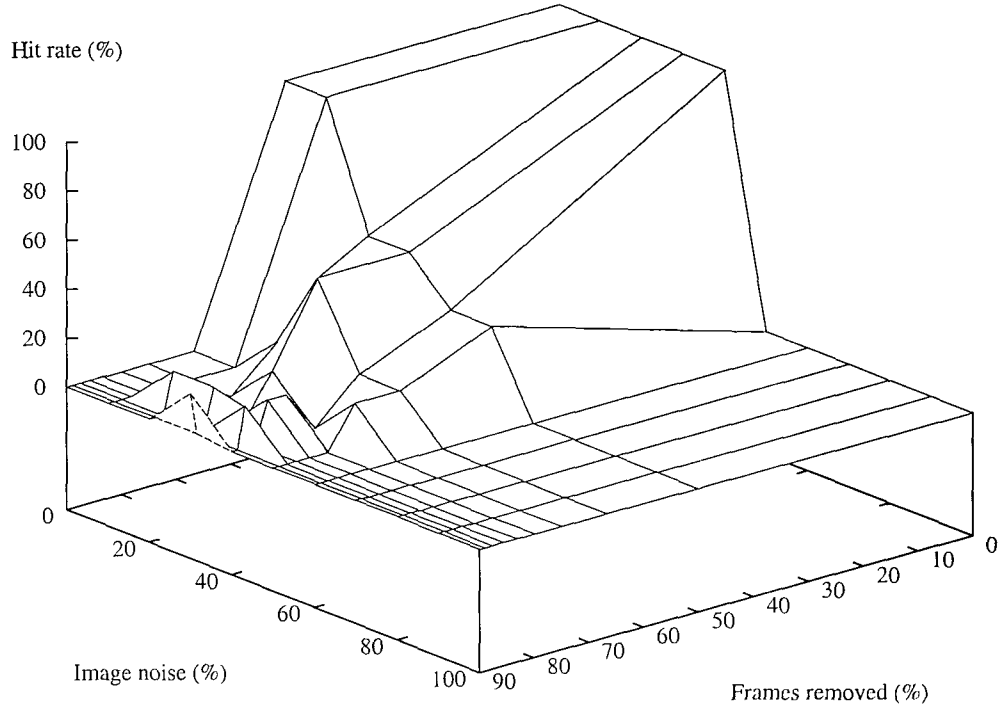
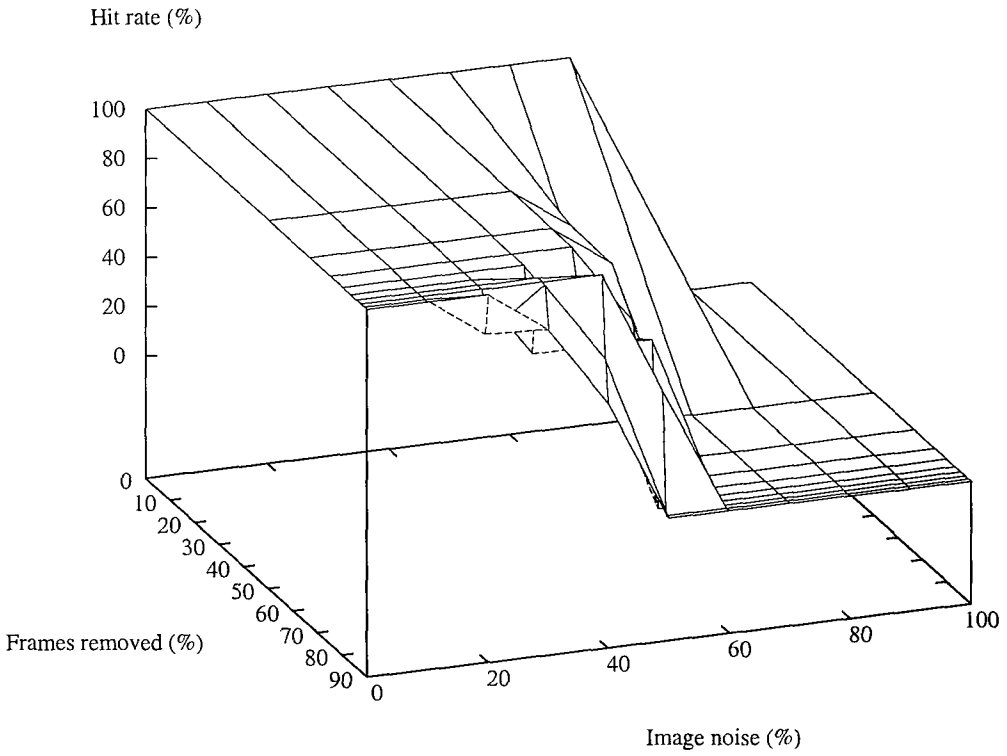


Figure 16: Two views of CVHT performance in simulated time-lapse imagery with varying levels of noise.

The graphs in Figure 16 show the performance of the CVHT for various levels of image noise on top of the occlusion caused by the time-lapse simulation. With the additional effect of noise, the CVHT's performance is affected much more quickly by the time-lapse occlusion with total collapse reached at 50% image noise for all levels of occlusion and partial failure occurring earlier at lower occlusion levels than the tests without noise.

An interesting effect occurs in the 20-50% image noise, >70% occlusion band where the CVHT begins to correctly detect the target in conditions where, with low image noise, it previously failed. It is probable that the occlusion has, by chance, removed some frames that caused problems for the full sequence extraction. Due to the linear motion model attempting to represent a non-linear motion, it is likely that there are significant portions of the sequence that qualify as "bad" frames. Removing some of these troublesome frames brings the algorithm to a close balance where the hit rate was only just 0%. The further addition of image noise then tips this balance towards a low hit rate. This conjecture is supported by the graphs in Figure 17 below, which are identical to those above but show near misses instead of just direct hits. A near miss is defined as a peak in the accumulator that is less than approximately three pixels (within an Euclidean distance of 3) from the correct peak. This includes direct hits. In the transition area discussed above (the 20-50% image noise, >70% occlusion region), the algorithm can be seen to be returning near misses as anticipated.

In summary, the CVHT can handle reasonable levels of occlusion and noise. Beginning when the occlusion destroys more than half of the data, the algorithm's performance enters a graceful decline, returning mostly near misses and occasional hits. Only when the majority of the data is consumed by the occlusion does performance collapse completely.



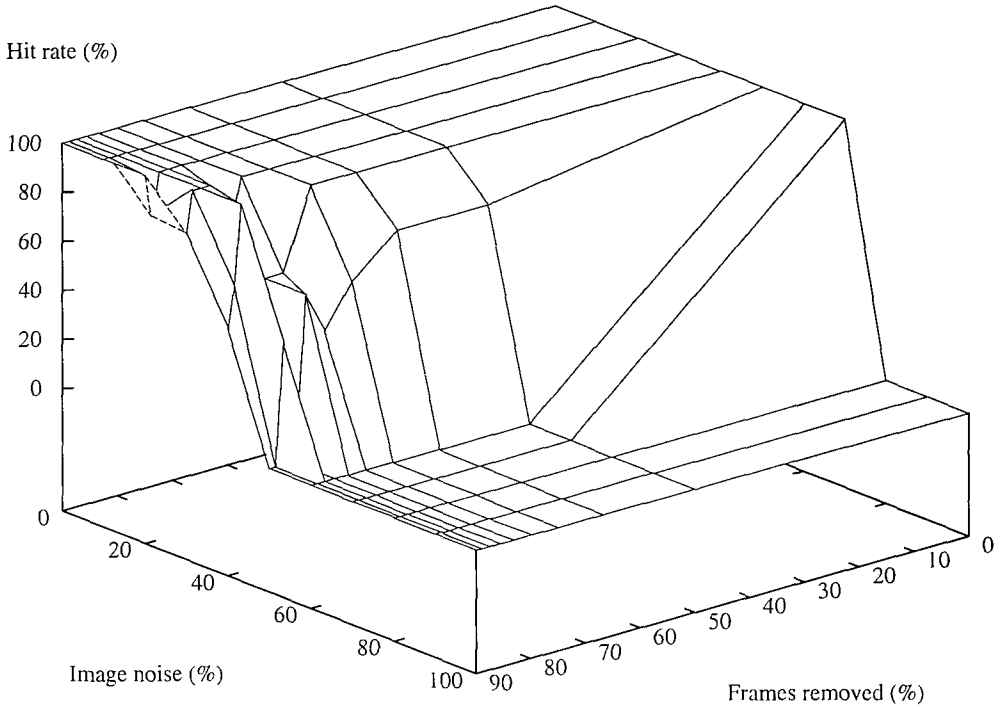


Figure 17: Two views of near miss performance in a simulated time-lapse sequence with varying levels of image noise

3.4.6 Real-world imagery: alternative motion models (Shuttle)

To illustrate the feasibility of alternative motion models, we have used the CVHT to extract parameters describing the launch of a Space Shuttle. Since we only consider the initial part of the launch phase, a linear velocity and acceleration model can approximate the movement. The sequence analysis begins on ignition of the rocket in order to avoid introducing extra parameters (higher-order differentials of acceleration, e.g. change in acceleration over time, etc) describing the pre-ignition phase where there is no acceleration at all. Despite this, the Shuttle appears to be static for the first three frames and then becomes airborne in the fourth. In addition to travelling upwards, the Shuttle also slips slightly to the right as it takes off. Whether this is a deliberate part of the trajectory or merely a minor deviation on launch is unknown.

The location, velocity and acceleration parameters of a booster during launch were correctly extracted by the new technique. Figure 18 and Figure 19 show the launch sequence with the correctly extracted booster highlighted in white. Interestingly, there should be a second significant peak in the accumulator at $y\ velocity = 0$ due to the slow initial take-off phase. This was not looked for at the time – only the largest peak was examined.

Whilst this example is empirical proof that the motion model is changeable, the CVHT is still limited by its parameterised nature. If the Shuttle sequence had included its full parabolic trajectory, this more complex motion would have to be incorporated into the accumulation phase, requiring many more parameters and possibly becoming computationally impractical.

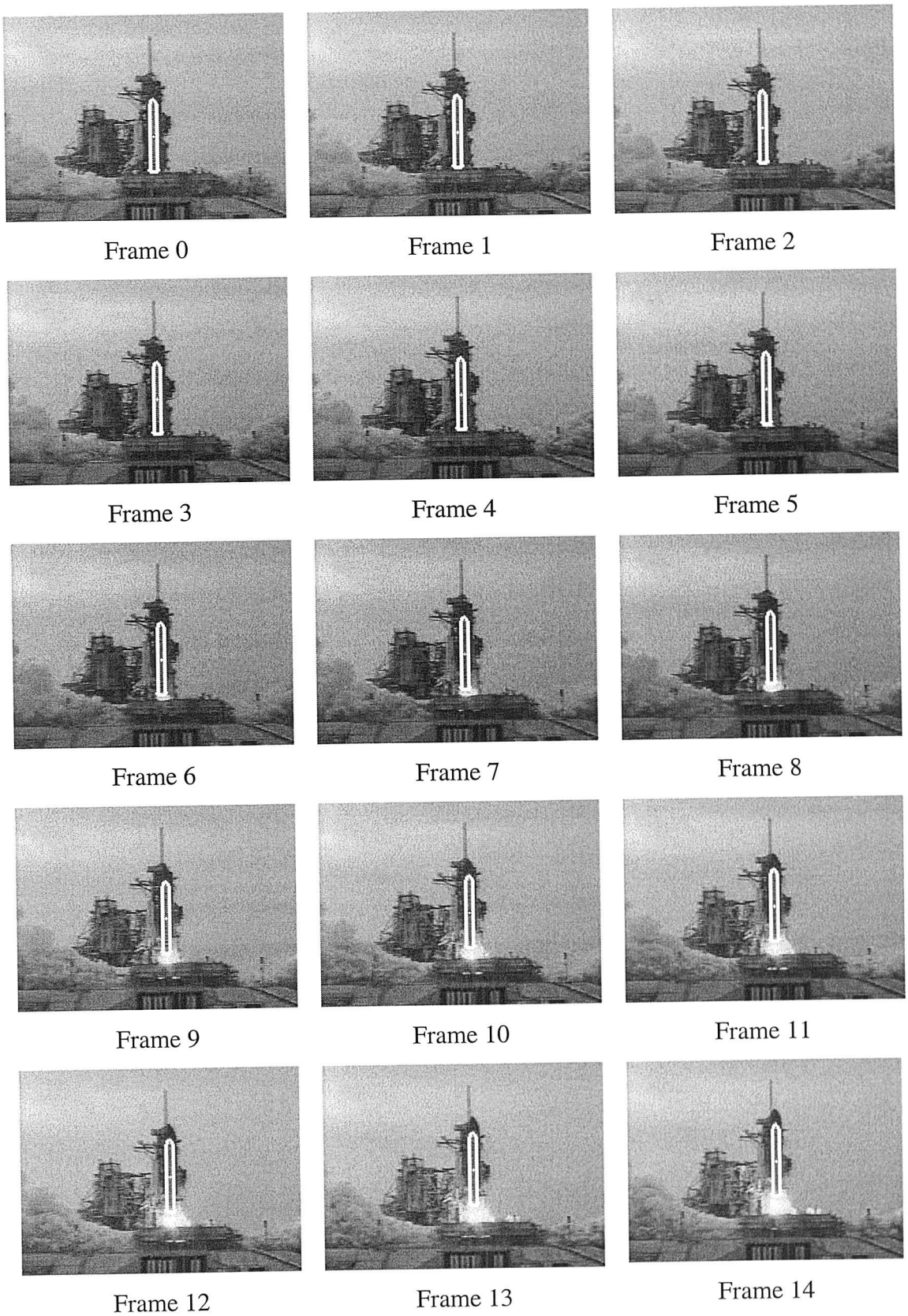
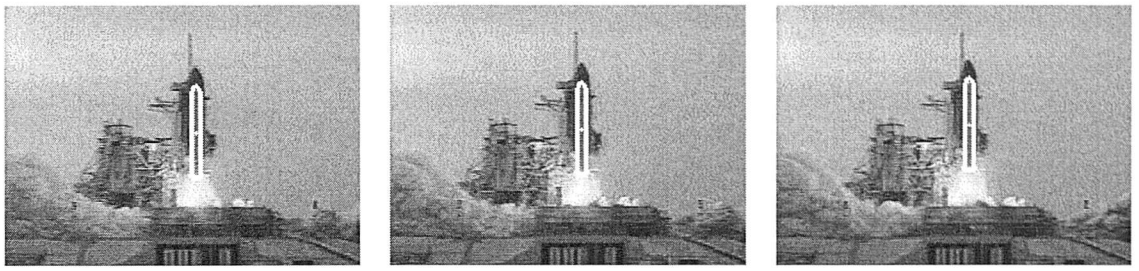
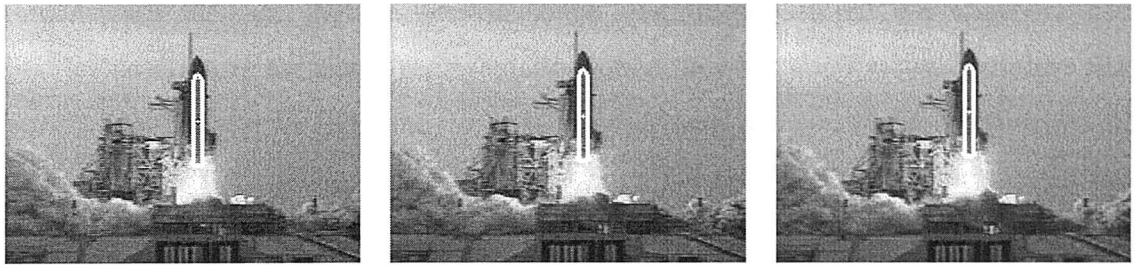


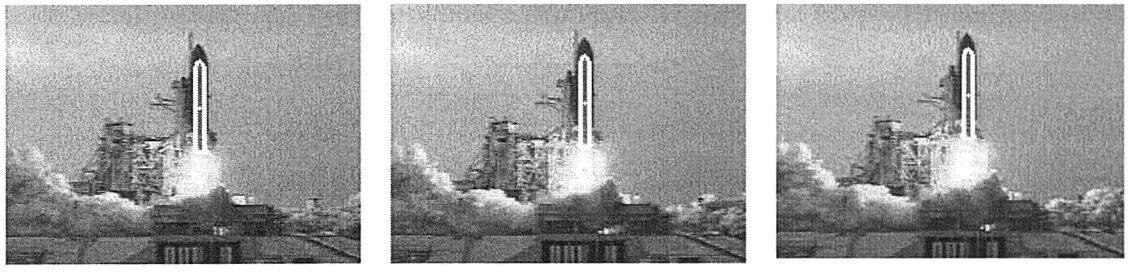
Figure 18: (part 1 of 2) Space-Shuttle launch with extracted template (of the booster) superimposed in white



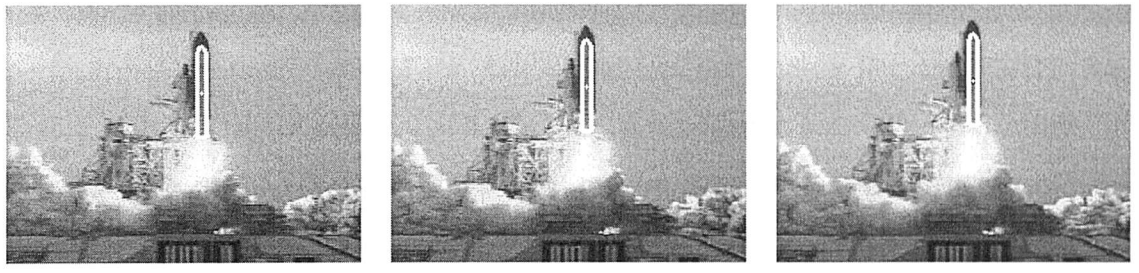
Frame 15 Frame 16 Frame 17



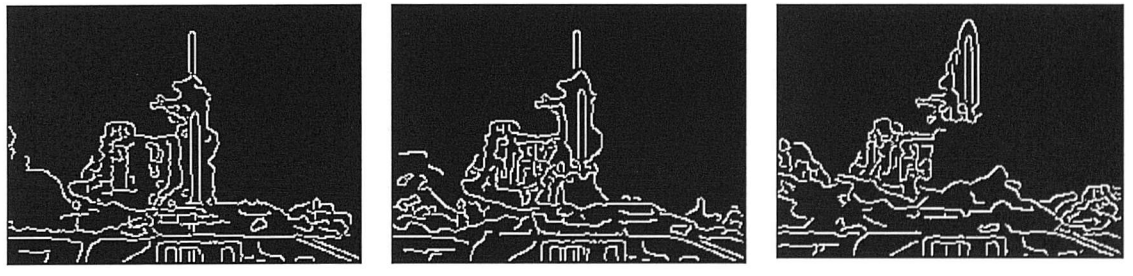
Frame 18 Frame 19 Frame 20



Frame 21 Frame 22 Frame 23



Frame 24 Frame 25 Frame 26



Edge-detected frame 0 Edge-detected frame 13 Edge-detected frame 26

Figure 19: (part 2 of 2) Space-Shuttle launch with extracted template (of the booster) superimposed in white, and some edge-detected example frames

3.5 Conclusions

In this chapter, we have developed the concept of the VHT as a moving object extraction algorithm, adding the capability to efficiently detect arbitrary shapes. This additional capability is provided by using a shape template - avoiding the exponential dimensional increase that parametric shape description brings. We chose to use Fourier descriptors for shape representation since they are continuous, complete and provide easy access to frequency content; furthermore, they have been proven in an earlier single-frame technique. The VHT brings the following characteristics to the union of techniques: it utilises temporal correlation to underpin evidence-gathering across a sequence, it does not call for initialisation or training and it employs the HT to implicitly solve the correspondence problem. Theory has been expounded and an evaluation made of the resulting implementation.

A comparative study of the new technique with a GHT-based frame-by-frame approach showed significant improvements in accuracy of extraction, particularly in conditions simulating high noise or occlusion. These improvements result from the reduction of quantisation noise (owing to continuous shape models) and integration of the whole sequence in the accumulation phase. Further experimentation explored performance characteristics of the new algorithm, concluding that the CVHT was usable with real-world imagery and (simulated) time-lapse video. For verification of the flexibility of the CVHT, an alternative motion model was implemented and used to correctly extract a booster from a launch sequence of a Space Shuttle.

In conclusion, adding arbitrary shape extraction (using a continuous representation) to the VHT has extended the generality of the algorithm without compromising its original robustness. In fact, using a continuous shape representation is an improvement on the robustness offered by a non-analytic (GHT-like) representation. The link between shape description and accumulator parameterisation has been broken by using templates, avoiding massive computational costs that are an inevitable consequence of extracting complex shapes with the VHT. However this dependency remains with motion description - the cost of the algorithm is still directly tied to the complexity of the motion model.

4 Motion Templates

4.1 Discussion

The earlier approaches to moving shape extraction are limited by the nature of the parameterised motion model - specifically its high dimensionality and hence reduced practical descriptive capability. For a complex motion, an explicit parameterisation system leads to an accumulator space approaching infinite dimensionality, a problem earlier alleviated in shape extraction by the use of shape templates. Following this earlier example, the motion of the shape could also be stored in template form - i.e. using motion templates to describe the movement of the target. Such a change removes the dimensionality limitation and enables the extraction of arbitrary shapes undergoing arbitrary motion. Storing motion information in this way is a form of a temporal template. Temporal templates are a technique for representing the movement of bodies through a sequence of images by encoding a motion trajectory. The encoding takes many forms - for example, there are many algorithms that combine spatial and temporal information into an XYT space to enable detection of particular movement patterns (e.g. detection of repetitive motion using temporal textures [50]). Other instances of the temporal template technique include a neural network based human motion tracker [61] that combines positional displacements with spatial templates of a human contour. The system contains several of these state vectors and is capable of tracking and predicting transitions between them. Similar efforts have been made using dynamic programming to track state transitions in gesture recognition [9].

With these templates it is no longer necessary to accumulate for the parameters describing the motion since, clearly, they are already known. It may be useful to imagine the motion template as an infinite dimensional parametric motion model where all the parameters have a fixed value. Naturally, the motion templates require *a priori* knowledge concerning the target object's path before analysis can begin. Since, by definition, tracking precludes the possession of this information, it is important to observe that the niche for motion templates, in the context of evidence gathering, is extraction or recognition (an example application might be searching a video database). Motion templates make it possible to robustly and efficiently extract parameters describing a shape that is following a specified trajectory. It can even be

argued that a limited prediction capability exists if the motion template covers a larger time span than is analysed (or if the motion is repetitive) since it is assumed that the extracted target will continue to follow the specified path.

As a result of the requirement for detailed prior knowledge, the new algorithm will be of use in cases where the general path of motion is known (e.g. cars turning at traffic lights will follow roughly the same path). However, there may be difficulties in real world imagery since not all objects will follow exactly the same path. The forgiving nature of an evidence-gathering approach should abate this concern provided the deviations are not excessive. If they are, the voting or peak detection algorithms can be arranged to handle the uncertainty, (e.g. as in the Fuzzy HT [25]).

Motion templates must encode the relative position of the target object at all times. This will automatically describe properties of the motion such as speed, acceleration, change in acceleration, etc. Motion templates should also record changes in scale and rotation over time since many moving objects rotate (e.g. a car when viewed from above rotates as it corners) and scale (e.g. due to perspective effects). Note that this additional detail will not cause any increase in accumulator dimensionality – the complexity of the algorithm is the same, only the complexity of the motion template itself has increased. More information relating to time-structured changes in the model shape (e.g. changing shape models throughout the sequence to represent deformations) may be recorded, but position, orientation and scale (temporal and spatial) are a minimal base set. These basic parameters exist in the implementation used for this thesis, but have not been sufficiently tested to document here.

The representation of the motion templates should be continuous so as to avoid the problems of discrete representations (Section 2.2). As such, it seems prudent to use Fourier descriptors for both motion and shape templates since these descriptors are well understood and Fourier approaches can handle many situations (e.g. irregular path sampling). An additional advantage is access to frequencies in the motion template, which may be of use in certain applications (for example, gait analysis may benefit from this characteristic - see Section 5.1). Furthermore, we have a consistent framework for the representation of both arbitrary shape and motion.

To encode the path for input to an implementation, it is convenient to specify a series of waypoints to encode the path, rather than use a smooth and complete description of the motion. The representation chosen must be able to take this data and interpolate it in a smooth fashion. Fourier descriptors have been designed to do

this and only need minimal modification to work with variable time periods between waypoints (required since movements may be quick or slow). One problem with this approach lies in under-sampling the path with too few waypoints. In this case, it is possible to over-fit the FDs and reproduce this under-sampled path too exactly. If desired, the waypoints may be filtered to generate a smoother-flowing path.

The introduction of motion templates into the HT requires no additional parameters to be searched for in the accumulation phase beyond the standard shape deformation parameters. However, such an inflexible implementation would excessively restrict the functionality. We consider that the essential parameters are rotation of the motion template in its spatial dimensions and scaling in both spatial and temporal dimensions. The scaling in the spatial axes does not need to be independent (i.e. it can be uniform scaling) since, we will consider only affine transformations initially. Scaling in the temporal axis adjusts the time taken to traverse the motion template and thus the speed with which an object must move to be identified as the target. Using the previous example, this would allow the algorithm to locate cars that are cornering either quickly or slowly. Finally, we must add an offset or phase parameter to separate the time encoded in a motion template from that used in a sequence. This is a temporal offset that corresponds with the spatial offsets already present in the HT (x and y position in an image). Without an offset parameter, time zero in the motion template must equate to frame time zero in the sequence; consequently, the algorithm would be unable to correctly extract parameters for a sequence that began with the target object part-way through its trajectory. Hence, the time-scale parameter provides temporal scale invariance and the offset parameter gives temporal translation invariance. Rotation invariance is not required since time is one-dimensional.

One probable candidate parameter that has not been included is a "direction" flag, indicating whether the motion template should be traversed forwards or backwards (a two-state variable, which would therefore doubling the parameter space size). This was omitted partly because the direction of motion was known in our test cases and partly because it is simple to reverse the motion template and re-run a test, thus avoiding additional complications in implementation.

Accordingly, the use of motion templates adds four extra parameters to the four required for shape extraction (as in the CVHT), giving in total an eight–dimensional accumulator. While this will cause a large increase in the computational requirements,

it is much less than the dimensional explosion threatened by polynomial extension. Clearly, the ability to extract, optimally and robustly, arbitrary shapes following an arbitrary path is well worth the additional computational resources.

4.1.1 Computational cost

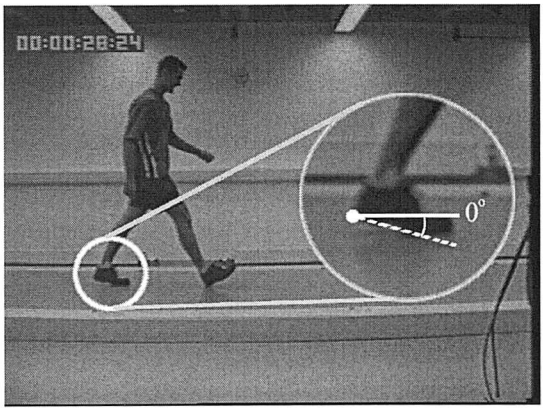
The algorithm is of order $O(N^8)$ in terms of complexity or $O(\#points \times \#s \times \#r \times \#mt\_s \times \#mt\_r \times \#t\_off \times \#t\_s)$ in terms of the number of operations required, where *#points* is the number of feature points (e.g. thresholded edge-pixels) in the sequence and *#s*, *#r*, *#mt_s*, *#mt_r*, *#t_off* and *#t_s* are the number of discrete steps in the parameter ranges for initial shape scale and rotation, scale and rotation of the motion template and temporal offset (phase) and scaling respectively. Hence, the new approach inherits the usual computational cost penalty of the HT: the accumulator is eight-dimensional and can require significant resources. However, most of the speed-up and memory-reduction modifications to HT-related techniques [38] are applicable – see Section 2.4.

As before, the complexity can be split into a shape and motion components. Like the CVHT, this algorithm has a fixed cost shape description - $O(N^4)$ – but, unlike the CVHT, it also has a fixed cost motion description - again $O(N^4)$. The crucial difference is that the complexity of the motion description part is fixed for any type of motion, thus giving better scaling of the algorithm to real-world situations.

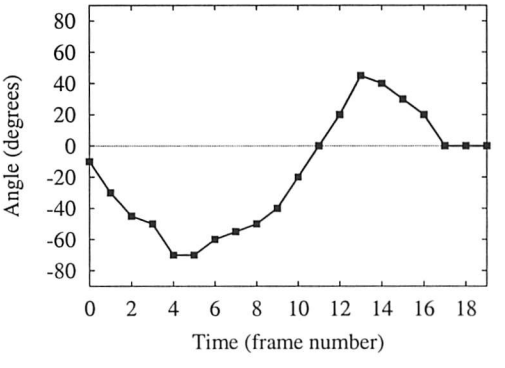
4.1.2 An example motion template

As an example of the information that may be encoded in a motion template, Figure 20 shows the changes in the x and y positions and the angle of the left foot of the walker, pictured as he walks across the sequence. This is a simple case showing only three components of the encoded data. A more direct (or unified) visualisation of a motion template than that shown in the component graphs below is difficult to produce because it must encode multiple dimensions of data (e.g. x, y, rotation, time), not to mention the possibility of the path looping back on itself. This overlapping of the path only occurs when one reduces the dimensionality of a motion template for visualisation purposes. It is not possible for this to occur in the motion template itself because there can only be one sample for any point in time, hence all points can be uniquely distinguished. The use of an additional dimension to separate otherwise interfering components is a well-established technique (e.g. [9]).

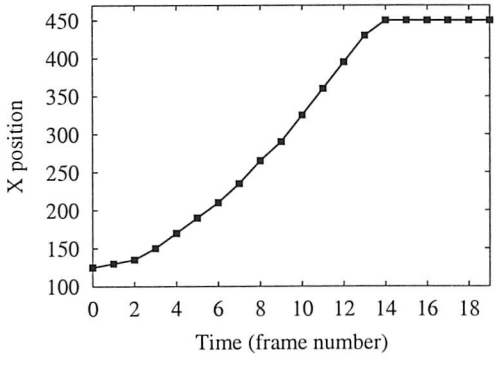
The measurements were made manually with the x and y positions being recorded for the rear of the heel of the left foot and the angle being estimated by eye (where a negative angle indicates the heel is up and the toe down). This illustrates the information that a motion template might have to record, in this case for extracting the location of a foot (a better description of the gait cycle may be found in [42]). Figure 20a shows the initial frame of the twenty-frame sequence that was examined. Figure 20c and Figure 20d contain the x and y components, which behave as expected - x increases as the foot swings forward and plateaus when it is placed on the ground so that the right foot can be raised; y goes through an arc as the foot is raised, swung and makes contact, then remains steady whilst the foot is pivoted upon. More interestingly, the angle component, Figure 20b, follows a more complex trajectory as the foot goes up onto tiptoe for the push-off phase of the gait cycle, follows a slow rotation until heel impact at frame 14 and quickly reverts to level with the floor as the weight of the body pushes the foot down.



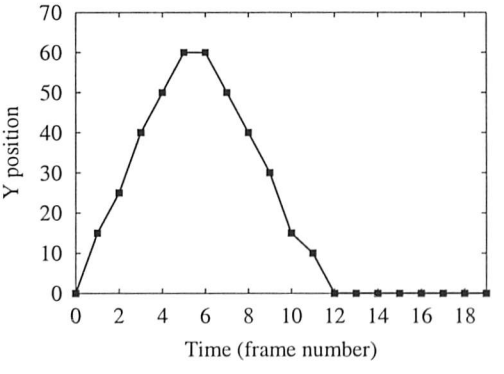
(a) Initial frame



(b) Motion template angle component



(c) Motion template *X* component



(d) Motion template *Y* component

Figure 20: Changes in x and y positions and rotation of a walker's foot angle

These three measures could be collected together to describe the anticipated motion of the foot. This then becomes the motion template that, when combined with a spatial template of a foot, would support extraction of human feet from a sequence of images.

In the person-extraction results presented below, we have continued to use the static torso rather than utilising the more advanced capabilities of the motion template algorithm to search for legs or other parts of the body. This has the advantage of retaining the comparability with earlier results and also avoids the issues of articulation and difficult edge extraction around the legs.

4.2 Theory

Before the theory, it may be helpful (as an aid to visualisation) to have a brief description of the processes performed by the motion template algorithm. For a given feature point (e.g. an edge pixel in a frame), a locus of points is plotted through the eight-dimensional accumulator space. This locus is constructed from the shape and motion descriptors, which are used to create transformed (scaled, rotated and translated) instances of the template shape. The transformations are intended to compensate for the expected motion of the object relative to the time reference of each frame and the anticipated scale and orientation of the object. Each of these instances is then traced into the accumulator in the two-dimensional x-y plane appropriate for the values of the various parameters. Once the voting process is complete, peaks in the accumulator indicate the location, rotation, scale, etc (at time $t = 0$) of an instance of the target shape moving along the specified path.

The analysis in this section is derived from the CVHT theory (Section 3.2), so there are similarities between the theory supporting motion templates and that of the CVHT. As one might expect, the changes to the CVHT theory are confined to implementing motion templates as the motion model. Hence the shape parameterisation is unchanged, so Equations 1 through 5 in Section 3.2 describe the shape template as before. Having parameterised the shape template, we must now parameterise the motion template, encoding the changing position, rotation and scale parameters. There are two major differences in the definition of the motion template from that of the shape parameterisation. The first is that the DC terms of the FDs are retained and used in the reconstruction. Removing them would effectively translate reconstructed co-ordinates relative to some arbitrary origin (i.e. the centre of mass of

the template), which would destroy the utility of the sequence by moving the start point. With motion templates, the start point is crucial to proper motion backtracking. Secondly, and relevant in terms of implementation or specifics of template definition only, the offsets used are not derived from chain codes, which have fixed x, y and t values, but come from a list of waypoints and thus can instead specify larger increments.

Here we recapitulate the FD definitions, but adapted to representing the motion template: A curve defined by two orthogonal components $MT_x(T_{ref})$ and $MT_y(T_{ref})$, representing the motion template MT, is parameterised by a normalised time reference $T_{ref} \in [0,2\pi)$ as follows:

$$a_{xk} = \frac{1}{\pi}\int_{-\pi}^{\pi} MT_x(T_{ref})\cos(kT_{ref})\mathrm{dT} \quad \text{and} \quad b_{xk} = \frac{1}{\pi}\int_{-\pi}^{\pi} MT_x(T_{ref})\sin(kT_{ref})\mathrm{dT} \quad (12)$$

with matching equations for the y component, where k is the harmonic number. As before, the larger k is, the more accurate the representation will be (up to some limit defined by the application scenario). Co-ordinates are recovered from the Fourier description using:

$$m_x(T_{ref}, \overline{MTFD}_x) = \sum_{k=0}^{n}\left(a_{xk}\cos(kT_{ref}) + b_{xk}\sin(kT_{ref})\right) \quad (13)$$

where $\overline{MTFD}_x = \{a_{x1}, b_{x1}, a_{x2}, b_{x2}, \ldots, a_{xn}, b_{xn}\}$ (the FD components of MT), again with a matching equation in y. Note the inclusion of the DC component ($k=0$).

So, in terms of theory, the motion templates are essentially identical to the shape representation FDs. In terms of implementation, there is the one significant difference mentioned above. In the shape description, the curve is piecewise linear and described with chain codes prior to Fourier encoding; in contrast, the waypoint structure describing the motion template is not an eight-way neighbour-connected chain but more of a directed graph. Each waypoint is specified by intermediate links detailing how far in the x, y and t dimensions to move in order to reach the next point – chain codes always indicate a movement of 0 or 1 units in x and y, and, in Kuhl's theory [35], the change in t is set by the assumption of a fixed traversal speed and the Pythagorean length of the combined x and y components. In adapting Kuhl's theory of elliptic Fourier Descriptors, the change in x or y (Δx or Δy) has become variable rather than fixed to 1 or $\sqrt{2}$ and the change in time (Δt) is user selectable (thus allowing variable speeds at different points on the path). This breaks Kuhl's constant speed

assumption but causes no difficulties with the rest of the algorithm because this assumption was only used to calculate Δt for particular Δx or Δy values.

We now have $m_x(T_{ref})$ and $m_y(T_{ref})$ (assuming an implicit $\overline{MTFD}$ parameter), which are functions that decode a set of motion template FDs to recover a co-ordinate (for x position and y position respectively) for a particular frame time T_{ref} (normalised to 0 to 2π for FD reconstruction). In addition, let $m_\rho(T_{ref})$ and $m_l(T_{ref})$ be functions that take a frame time T_{ref} and recover rotation and scale co-ordinates. As orthogonal components (i.e. with no direct relation), these are defined in the same way as $m_x(T_{ref})$ and $m_y(T_{ref})$ above.

The composite entity created by using these co-ordinate recovery functions together gives us a motion template. This motion template is now used to transform the co-ordinates calculated from the shape descriptors. These co-ordinates have already been globally scaled and rotated (i.e. we are using R_x and R_y from Equation 5) to adjust for possible initial scales and orientations. They are then further scaled and rotated according to the stored values in the motion template, thus allowing for changes during motion:

$$\begin{aligned}\xi_x(s,T_{ref},T_{off},\mathbf{a}_s) = {} & m_l(T_{ref}-T_{off})R_x(s,\mathbf{a}_s)\cos(m_\rho(T_{ref}-T_{off})) \\ & -m_l(T_{ref}-T_{off})R_y(s,\mathbf{a}_s)\sin(m_\rho(T_{ref}-T_{off}))\end{aligned} \tag{14}$$

with a similar equation for y co-ordinates and where s is a free variable specifying a point on the shape template and $\mathbf{a}_s$ is a vector of the shape template's rotation and scale parameters. T_{ref} is the time reference (frame number) of the current image and T_{off} is a phase parameter that offsets any mismatch between the frame times in the sequence and in the motion template. $m_l(T_{ref} - T_{off})$ and $m_\rho(T_{ref} - T_{off})$ recover scale and rotation, respectively, for time $T_{ref} - T_{off}$ from the motion template. These rotated and scaled co-ordinates now represent the shape at the expected orientation and scale. Next, they are translated to compensate for the object's expected motion. However, the path of expected motion is also scaled and rotated, requiring parameters $\mathbf{a}_m = [l_m \quad \rho_m]$,

$$\begin{aligned}\mu_x(s,T_{ref},T_{off},\mathbf{a}_m,\mathbf{a}_s) = {} & \xi_x(s,T_{ref},T_{off},\mathbf{a}_s) \\ & +l_m m_x(T_{ref}-T_{off})\cos(\rho_m) - l_m m_y(T_{ref}-T_{off})\sin(\rho_m)\end{aligned} \tag{15}$$

again with a similar equation in y and where $m_x(T_{ref} - T_{off})$ and $m_y(T_{ref} - T_{off})$ are the x and y offsets recovered from the motion template at time $T_{ref} - T_{off}$. We can now form the kernel that defines the shape of votes in the accumulator. This is a multi-

dimensional combination of the template at a number of translations, orientations and scales, and can be obtained from:

$$\begin{aligned}\overline{\omega}(s,T_{ref},T_{off},l_T,\mathbf{a}_m,\mathbf{a}_s) = \mu_x(s,(T_{ref}\cdot l_T),T_{off},\mathbf{a}_m,\mathbf{a}_s)U_x \\ + \mu_y(s,(T_{ref}\cdot l_T),T_{off},\mathbf{a}_m,\mathbf{a}_s)U_y\end{aligned} \quad (16)$$

One final parameter, l_T, has been added to perform temporal scaling on the motion template, thereby allowing adjustment of the speed at which the path is traversed. To ensure that the locus drawn in the accumulator passes through the reference point, the kernel is offset from the image co-ordinates of each feature point. Hence, for an image sequence *IS* (first defined in Equation 7 but repeated here for convenience), the votes are placed in the accumulator as A_t:

$$IS = \{\overline{\lambda}(P,T_{ref}) \mid P \in D_P, T_{ref} \in D_T\} \quad (17)$$

$$A_{P,T_{ref}} = \{\overline{\lambda}(P,T_{ref}) - \overline{\omega}(s,T_{ref},T_{off},l_T,\mathbf{a}_m,\mathbf{a}_s) \mid s \in D_s\} \forall P \in D_P, T_{ref} \in D_T \quad (18)$$

As before, a suffix on the domain indicates its extent (here, D_P is the domain of an image in the sequence and D_T is the set of frame times in the sequence). $\overline{\lambda}(P,T_{ref})$ retrieves feature points from the sequence. Again, as before, the parameter space formed by the application of the expression above is mapped into an accumulator by the use of a matching function *M* (Equation 9). This multi-dimensional accumulator space is sampled into a discrete parameter space S_{DMT} given by:

$$S_{DMT}(\overline{b},l_T,T_{off},\mathbf{a}_m,\mathbf{a}_s) = \sum_{s\in D_s}\sum_{P\in D_P}\sum_{T_{ref}\in D_T} M(\overline{b},\overline{\lambda}(P,T_{ref}) - \overline{\omega}(s,T_{ref},T_{off},l_T,\mathbf{a}_m,\mathbf{a}_s)) \quad (19)$$

where $\overline{b}$ is a vector of the image *x* and *y* co-ordinates at time 0, l_T is the time scale-factor, T_{off} is the time offset (phase) and $\mathbf{a}_m$ and $\mathbf{a}_s$ contain scale and rotation parameters that respectively transform the motion template's path and transform the initial orientation of the target shape. This expression gives an accumulation strategy for finding arbitrary shapes moving arbitrarily. It allows extraction of the optimal parameters describing an arbitrary (but specified) shape of unknown orientation, position and scale that is following an arbitrary (but specified) path of unknown orientation and scale, which takes an unknown time to traverse.

The new algorithm currently traces the entire template shape in the accumulator for each feature point and for each parameter combination. The GHT places a restriction on which template points are drawn - only those with the same gradient direction as the edge pixel being processed are added to the accumulator so that only

the relevant fractions of the template are traced. With accurate gradient direction data, this restriction removes many unnecessary votes (and hence noise) from the accumulator. The voting algorithm may be changed to perform the same reduction of votes as in the GHT by incorporating a function that calculates the gradient direction at a point on the Fourier-described curve. This value would then be compared against edge pixel gradient direction to restrict votes cast into the accumulator. Depending on the computational cost of the gradient direction calculation, this reduction in voting may result in a linear speed improvement.

4.3 Pseudo-code algorithm

The algorithm that describes the motion template HT voting process is:

```
For (frame f of the sequence)
 For (edge pixels e_x, e_y exceeding threshold value)
  For (time_scale=timescale_min -> timescale_max)
   For (phase=phase_min -> phase_max)
    time=timescale * frame_time(f) + phase
    Recover m_x, m_y, m_p, m_l from MT(time)
    For (scale=scale_min -> scale_max)
     For (angle=angle_min -> angle_max)
      For (t=0 -> 2*PI)
       Generate vector from FD (t)
       Rotate vector by (angle + m_p)
       Scale vector by (scale + m_l)
       For (mt_global_angle= mt_global_angle_min ->
                             mt_global_angle_max)
        For (mt_global_scale= mt_global_scale_min ->
                              mt_global_scale_max)
         Rotate (m_x, m_y) by mt_global_angle
         Scale (m_x, m_y) by mt_global_scale
         Offset FD vector by (m_x, m_y)
         Offset FD vector from (e_x, e_y)
         Result is final co-ords (a_x, a_y)
         accum[a_x, a_y, scale, angle, timescale, phase,
               mt_global_angle, mt_global_scale] ++
```

4.4 Results

It is usually prudent to compare new techniques with contemporary or equivalent approaches. This appears unfeasible in this case since the most appropriate comparator technique is the GHT, but with interpolation, or tracking. The suitable form of this interpolation for the GHT (or the motion model for tracking) is actually the motion template, the very subject of these results. The alternative would be a fully representative parametric motion model that, as explained previously, is computationally intractable due to its infinite dimensionality. Consequently, the comparison that would be made is that of a frame-by-frame extraction process with a non-analytic template representation against an integrated multi-frame extraction process with an analytic template representation. This comparison has already been made in Section 3.4, which examined the earlier approach to moving arbitrary-shape extraction (without motion templates but with a linear motion model) and consequently fails to test the subject of this section - the motion template in an evidence gathering context.

Comparison with other techniques that use similar knowledge of motion (such as the neural network based human motion tracker [61] or the spatio-temporal repetitive motion detector using temporal textures [50] mentioned earlier) is not comparing like with like. Neither is comparison with other spatio-temporal based techniques (e.g. a snake that operates in a spatio-velocity space [49]) since they too are dissimilar at a core level. In the case of techniques that are as dissimilar as a tracker and an extractor, the comparison is best made on application-dependent qualitative requirements or on the basis of each technique's features (e.g. optimality vs. on-line performance), rather than a quantitative performance analysis. In light of these difficulties, we have examined the performance of the new technique in terms of noise affecting each component of the system rather than attempt to make direct comparisons with other distantly related approaches. We believe that such analysis will enable the aforementioned choice based on requirements or features.

The short note on pre-processing and noise models in Section 3.4.1 applies to this section also.

4.4.1 Image-noise performance on synthetic sequences

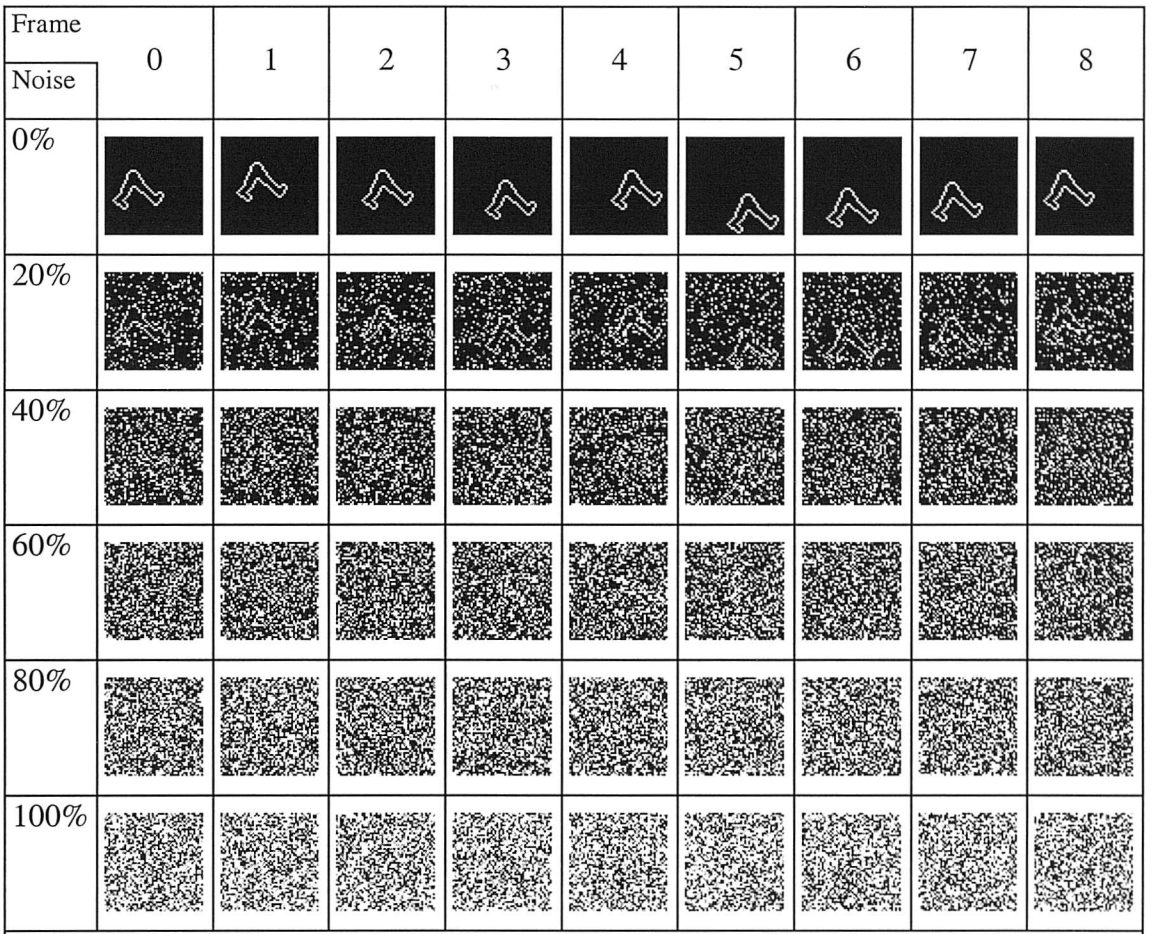
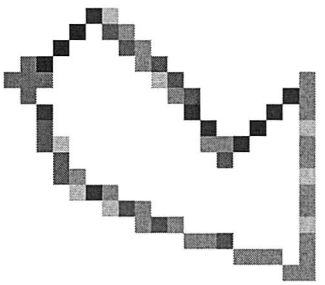


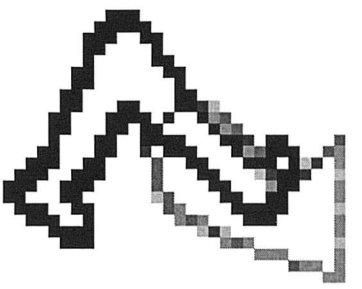
Figure 21: Full synthetic sequence with Gaussian noise

The new algorithm was run on a nine-frame synthetic sequence based on a small (50×50) image, Figure 21 (0% row), moving along the path shown in Figure 22a. The path was regularly sampled (in time) for this illustration and the grey-levels show the time taken to traverse each section of the path (the darker the pixel, the more time was spent traversing it; i.e. the slower the movement). The cross on the left edge of the motion template indicates the starting point with motion proceeding clockwise along the path shown. The motion template was given perfect co-ordinates since we are examining the response to image noise in the input sequence, not noise in the motion template. Again, a small image was chosen to make practical computation of large-scale tests. Noise (Gaussian wraparound) was added at random to each frame of the sequence at eleven noise levels from 0% random coverage to 100% random coverage of the frame, with pixel values wrapping rather than being clipped when the

addition of noise took the values out of range. The noise distribution was zero-mean Gaussian with a standard deviation of one. This noise function is harsher than that found in Section 3.4.1 because wrapping pixel values ensures that the data is changed - just thresholding an addition of noise to a white pixel may leave it unchanged. The noise function was altered when preliminary results using the original function indicated excellent performance at 100% noise! Examples of the effects of the increasing noise levels can be seen in Figure 21. The grey-level images produced are thresholded by the algorithm, although Figure 21 shows images prior to thresholding. Note that the shape is completely obliterated at the maximum noise level and that at around 50% noise it is nearly impossible to distinguish the shape by human vision.



(a) Motion template (small cross at far left is start point)



(b) Frame 0 of sequence with superimposed motion template

Figure 22: Motion trajectory of "legs" shape

Again, the new technique is shown to be capable of coping with significant levels of noise. The performance curve in Figure 23 is similar to those for previous VHT-derived techniques. This is accordance with earlier studies that found that the VHT-based techniques are able to handle noise levels that are approximately twenty percent greater than a comparable GHT-based frame-by-frame technique.

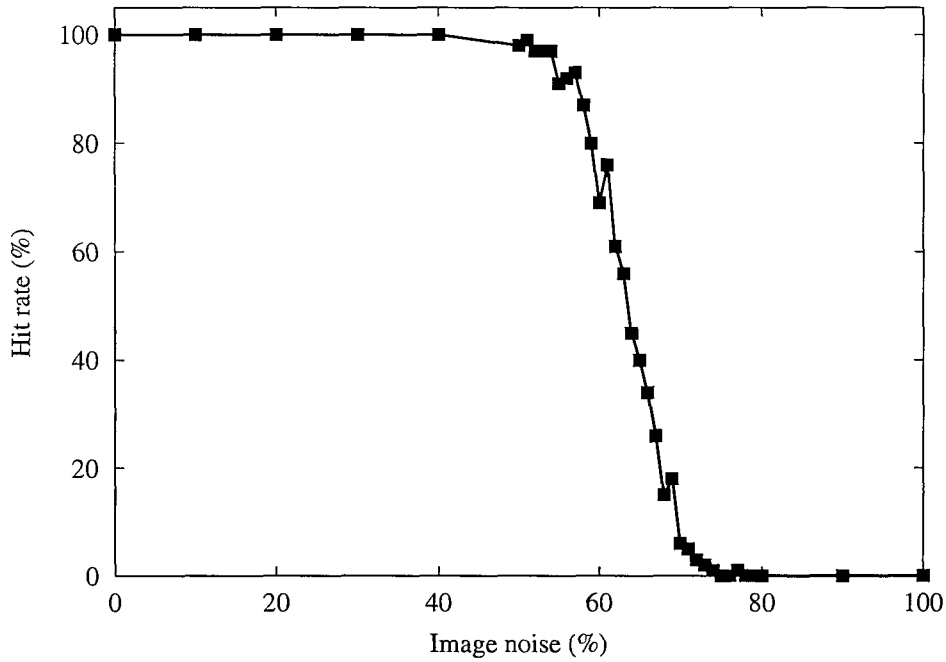
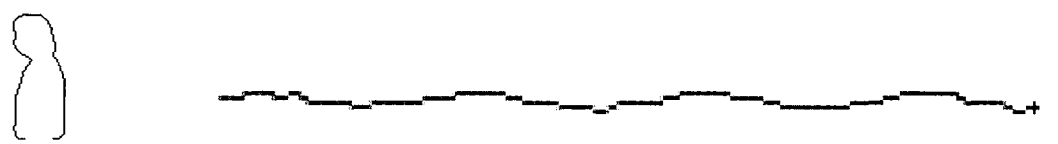


Figure 23: Noise performance for synthetic imagery

Contrasting these results with the earlier ones for the CVHT (Section 3.4.2) shows a similar but steeper performance curve. The CVHT began to lose accuracy at lower noise levels (40% versus 50%) and gradually declined to total failure at some point after 90% whilst the failure point for the motion template implementation is earlier at approximately 75%. On the face of it, this indicates the CVHT is more resilient but it must be noted that the noise function used here produces stronger damage to the edge-pixel data than that used for the CVHT tests. Above, we stated that using the motion template technique with the previous noise function returned excellent accuracy even at 100% noise. As explained above, the "clipped" noise function used for the CVHT tests may retain an edge-pixel at the correct level whilst the harsher "wrap-around" function guarantees to alter this information. So, with this taken into account, it suggests that the motion template implementation can tolerate significantly higher noise levels for similar conditions.

4.4.2 Image-noise performance on real-world sequences

Ultimately, the intention is to apply the new techniques to the analysis of human motion. For this purpose, and to substantiate the applicability to real-world imagery, we have evaluated the performance of motion templates when locating a walking person viewed from the side, as with the CVHT in Section 3.4.4. For each image in the sequence, co-ordinates specifying the particular motion of the walker were gathered by selecting a reference point on the body and estimating its position by eye. Since these measurements are likely to contain inaccuracies, the motion template itself is not perfect and will be another source of errors and peak-spread in the accumulator. Again, as with the CVHT, owing to the robustness of its formulation there is no need for exceptional precautions in the new technique.



(a) Torso and head template

(b) Motion template (begins at the small cross on the far right)

Figure 24: Shape and motion templates for CA1 sequence.

Again, we have used the torso rather than legs in these tests, despite the new technique being more capable of handling the more complex motion. This is in part because of the difficulties of extracting legs cleanly (self-occlusion and bad edge-detection due to shadows). More importantly, using the same part of the body allows us to retain the capability to make a limited comparison with the earlier CVHT tests. Figure 24 shows a reconstructed template of a walker's (CA1) torso and head, which was created by manual tracing from a typical frame in the sequence. Also shown is a plot of the x and y components of the motion template used. Figure 25-Figure 27 displays the frames of the walker sequence CA1 after a successful extraction; both the extracted shape and motion templates are superimposed on each image.

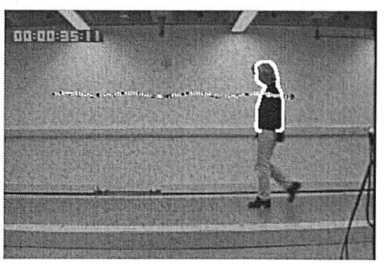
Frame 1

Frame 2

Frame 3

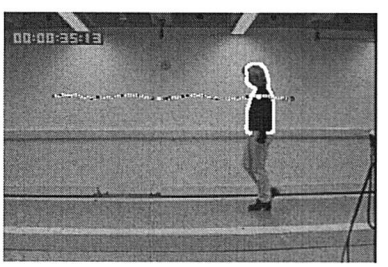
Frame 4

Frame 5

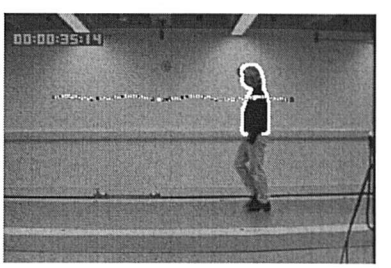


Frame 6

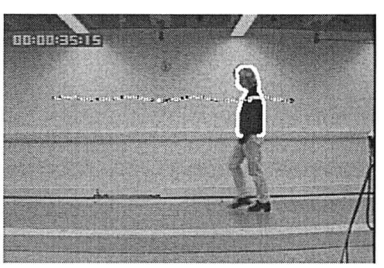
Frame 7



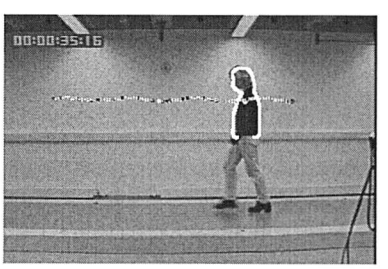
Frame 8



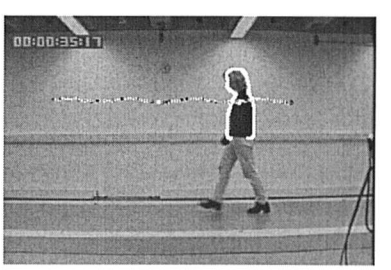
Frame 9



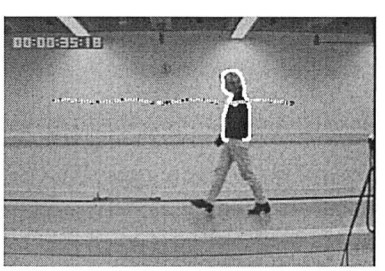
Frame 10



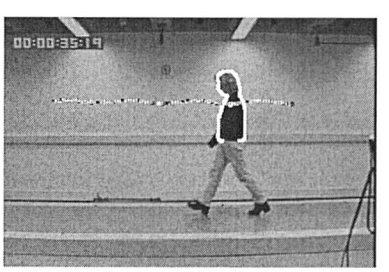
Frame 11



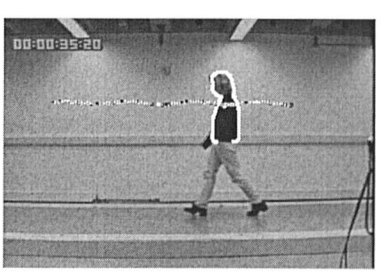
Frame 12



Frame 13

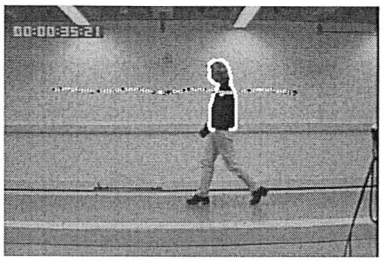


Frame 14



Frame 15

Figure 25: (part 1 of 3) Frames of sequence CA1 with superimposed templates.

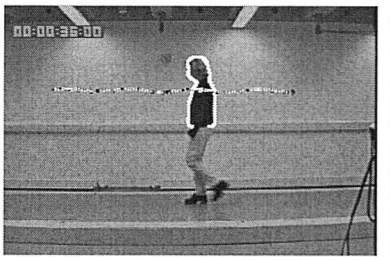


Frame 16

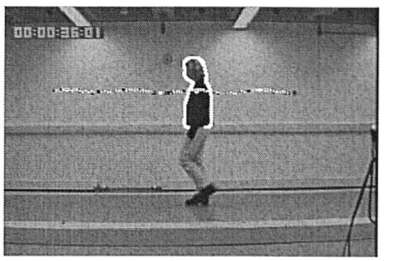
Frame 17

Frame 18

Frame 19

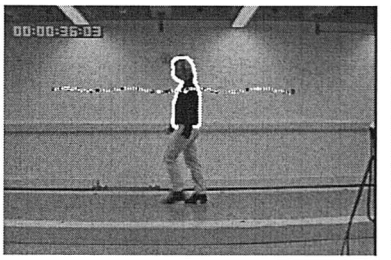


Frame 20

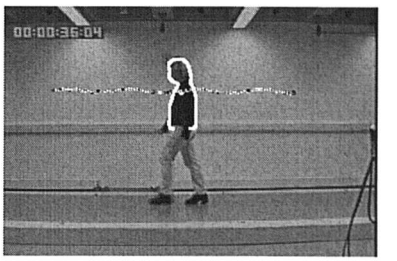


Frame 21

Frame 22

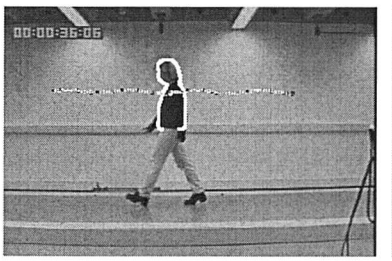


Frame 23

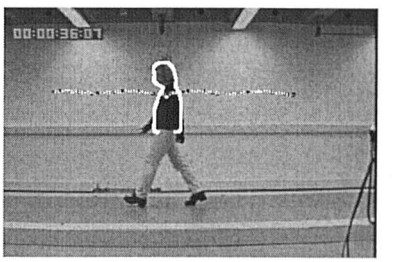


Frame 24

Frame 25

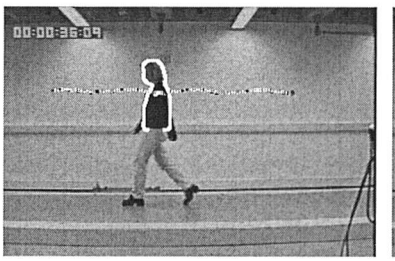


Frame 26

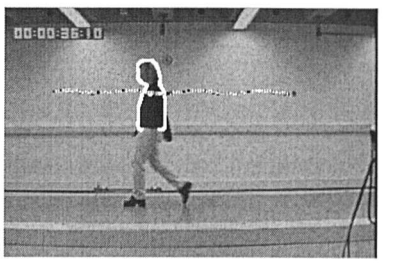


Frame 27

Frame 28



Frame 29

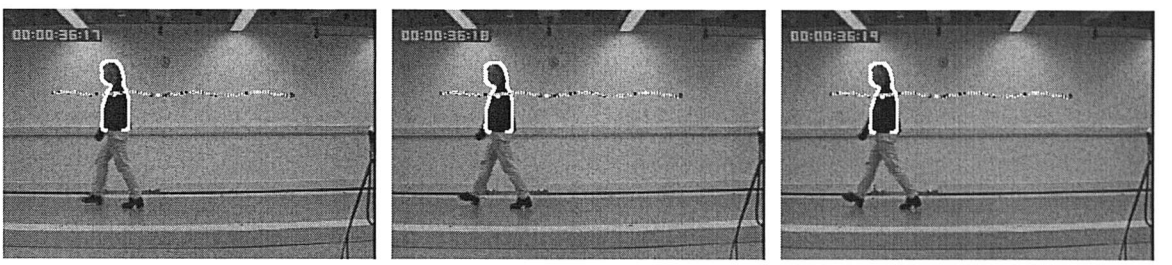


Frame 30

Figure 26: (part 2 of 3) Frames of sequence CA1 with superimposed templates.

Frame 31 Frame 32 Frame 33

Frame 34 Frame 35 Frame 36



Frame 37 Frame 38 Frame 39

Figure 27: (part 3 of 3) Frames of sequence CA1 with superimposed templates.

The noise model used in Section 4.4.1 (analysis of noisy synthetic imagery – wraparound Gaussian noise) was applied to each frame of the CA1 sequence, with examples of the different noise levels shown in Figure 13 (Section 3.4.4). The sequence processed here is identical both in content and in the type of noise added to it as that used in Section 3.4.4. Figure 13 shows the walker can be difficult to perceive in a single frame when the noise exceeds 40%. In fact, this is the point at which automatic extraction starts to fail, slightly earlier than for the synthesised imagery. The test results displayed in Figure 28 show the beginnings of accuracy loss at 40% noise, dropping off smoothly and missing the target completely above 80% noise. The difference between these results and those for synthetic imagery can be attributed to the imperfect conditions prevalent in the real world; e.g. the cluttered and noisy background and the imperfect shape and motion templates. Comparing it with the earlier CVHT result for this sequence, it can be seen that there has been a minor improvement. Since the CA1 subject walks quite "flatly", with minimal bobbing

motion, it seems that CVHT linear velocity motion model managed a good extraction result, similar to that of the motion template.

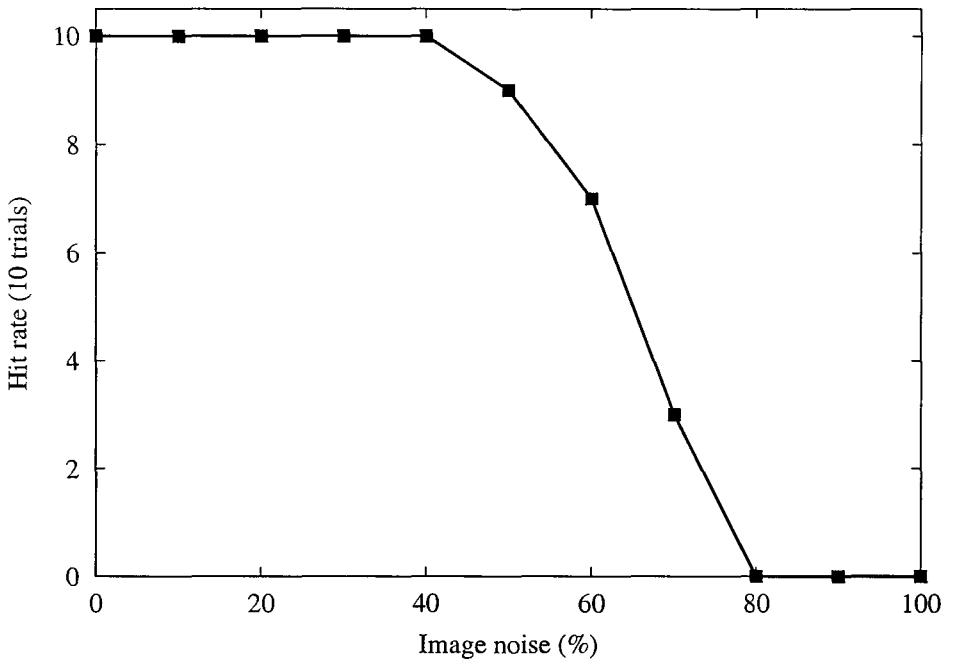


Figure 28: Extraction accuracy in increasing Gaussian noise for CA1 sequence

4.4.3 Image-occlusion performance on real-world sequences

A simple test of the effects of occlusion was carried out on the walker sequence CA1 described above. No noise was added to the sequence since this would be an unnecessary complicating factor. Instead, vertical lines of pixels were blanked out and the algorithm was run on the resulting images. The results revealed (Figure 29) that the new technique correctly extracts the walker until the blanking is 175 pixels wide – completely obscuring the walker for approximately 70% of the duration of the entire sequence. Furthermore (although this is not shown on the graph), the extracted peak is within one pixel of the true peak for another thirty pixels, indicating there is a measure of peak spreading. Like the CVHT, the new technique is capable of handling high levels of occlusion owing to the global integration of evidence across the entire sequence. This result is similar to the earlier CVHT synthetic occlusion tests (Section 3.4.3).

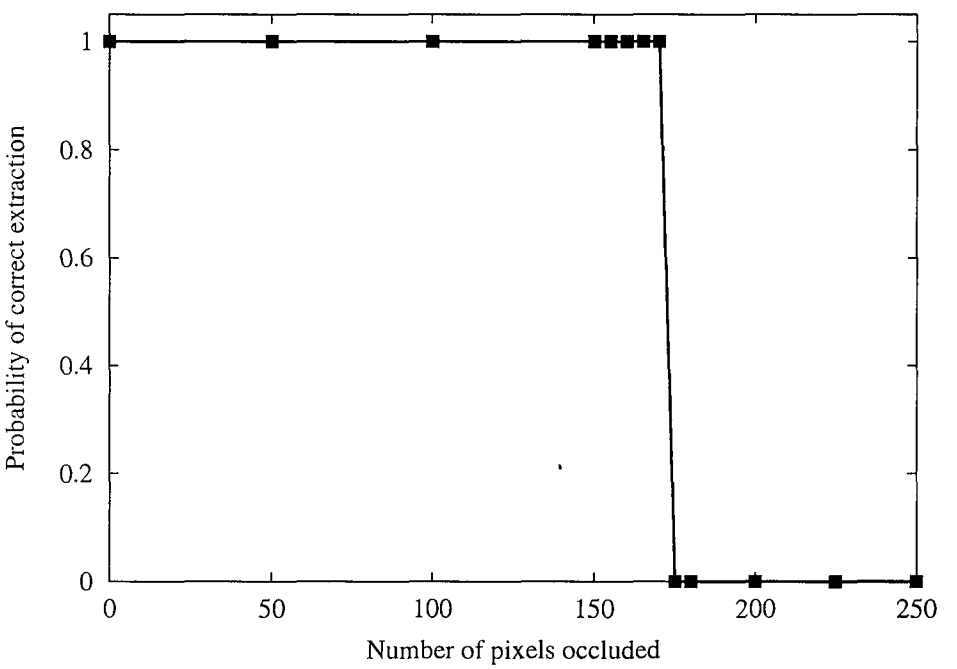
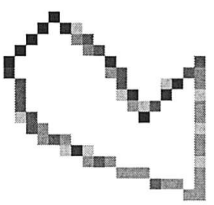


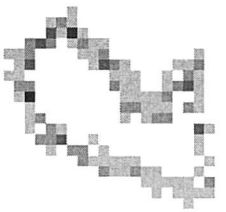
Figure 29: Occlusion tests on CA1 sequence

4.4.4 Effects of noise in the motion template

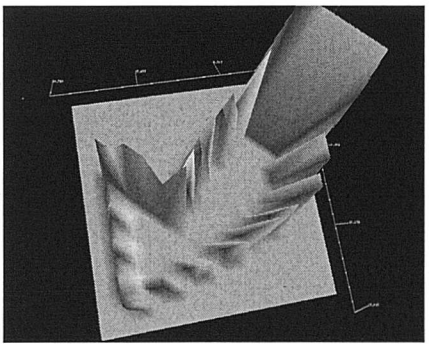
To determine the resilience to noise in the motion template, a percentage of co-ordinates calculated from it were perturbed by uniform noise. Note that adjusting these co-ordinates is equivalent to moving the target shape in the image sequence by the same amount. There are two dimensions to the noise: first, how many of the co-ordinates are affected; second, the maximum distance (in pixels) that would be added to each co-ordinate. These tests were performed on the synthetic sequence and motion template in Section 4.4.1. An example of this corruption can be seen in Figure 30 below (as before, the darker the pixel, the more time spent traversing that section of the path). The corrupted motion template displayed in Figure 30b and Figure 30d had a 100% probability of a co-ordinate being affected, with a maximum offset from the true path of 1 pixel.



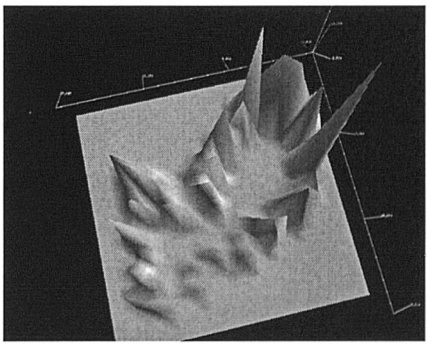
(a) Original, uncorrupted motion template



(b) Corrupted motion template



(c) 3D plot of original, uncorrupted motion template

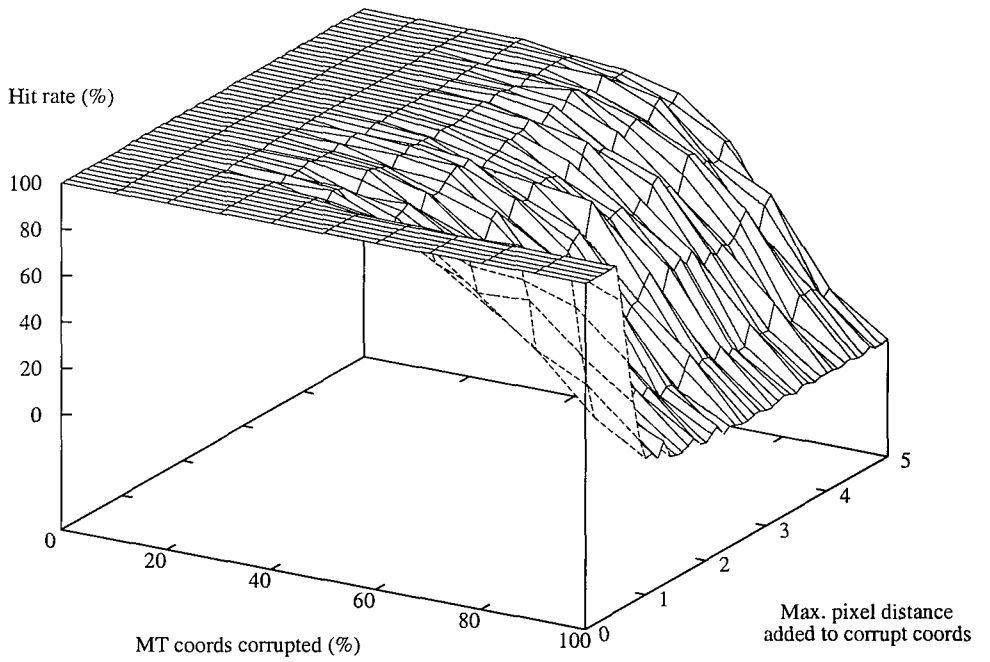


(d) 3D plot of corrupted motion template

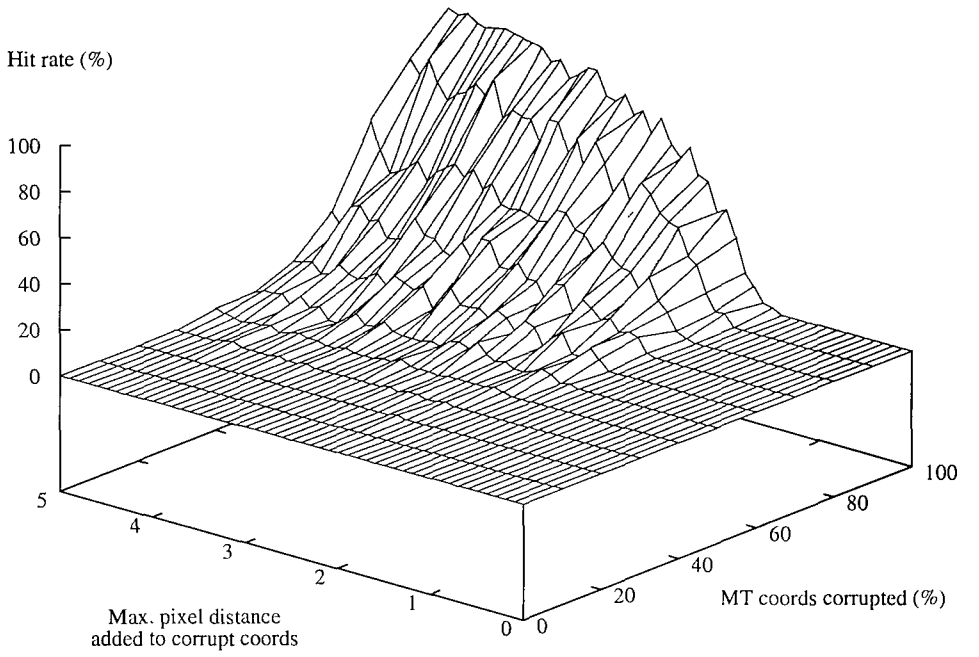
Figure 30: A corrupted motion template

The results show that as the percentage of co-ordinates corrupted increases, so the performance declines. Figure 31a shows the hit rate from 100 trials at each noise level, with a drop beginning after approximately 40% of motion template co-ordinates have been corrupted, and declining to total failure when all have been affected. The pixel distance added has no effect when its value is below 1 since quantisation in the accumulator removes any effect. Once the value is above 1, the effect cannot be negated and performance declines as described. Figure 31b shows the inverse of Figure 31a, the rising number of misses as a function of the two noise components. It is interesting to note that the drop-away in performance is fairly (although not completely) constant across the range of max-pixel-distances-added, perhaps because there is little competing noise in the synthetic sequence.

The mechanism for performance decline can be attributed to a "peak-spreading" effect common to all HT-derived techniques. As the accuracy of the input data decreases, the peaks in the accumulator become less defined (smaller and more spread out) and the background noise level rises. To begin with, the HT-algorithm will find the correct parameters but, as noise increases and the definition of the correct peak becomes smoother, the parameter estimates slip gradually from their true values. This continues until the spreading of the peak weakens it to the point where the algorithm is attracted to other potential sites. It is made clear that this is occurring by comparing the graphs in Figure 32a and Figure 32b, which show near misses (in the immediate neighbourhood of the target). At lower noise levels, the graphs show a constant hit rate, indicating that the output peak is within the ranges specified on the graphs, and demonstrating that slippage is occurring rather than a radically different peak location being selected. As the noise increases, the location of the peak moves by approximately the same amount as the pixel distance noise being added to the motion template. Ultimately, as the correct peak sinks into background noise in the accumulator, the algorithm's output moves to more distant and incorrect peaks.

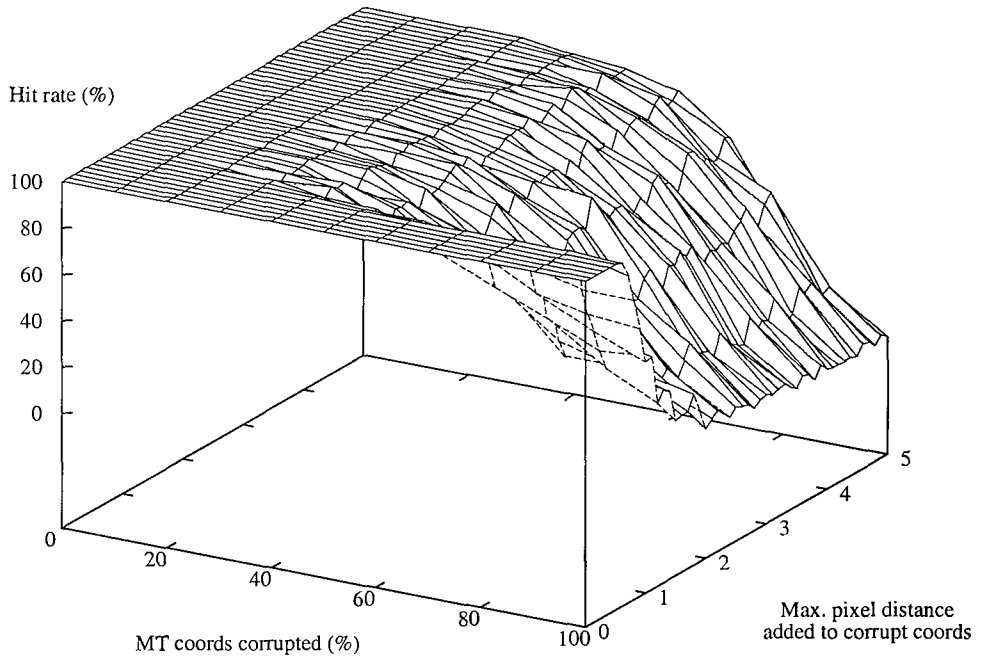


(a) Exact hits

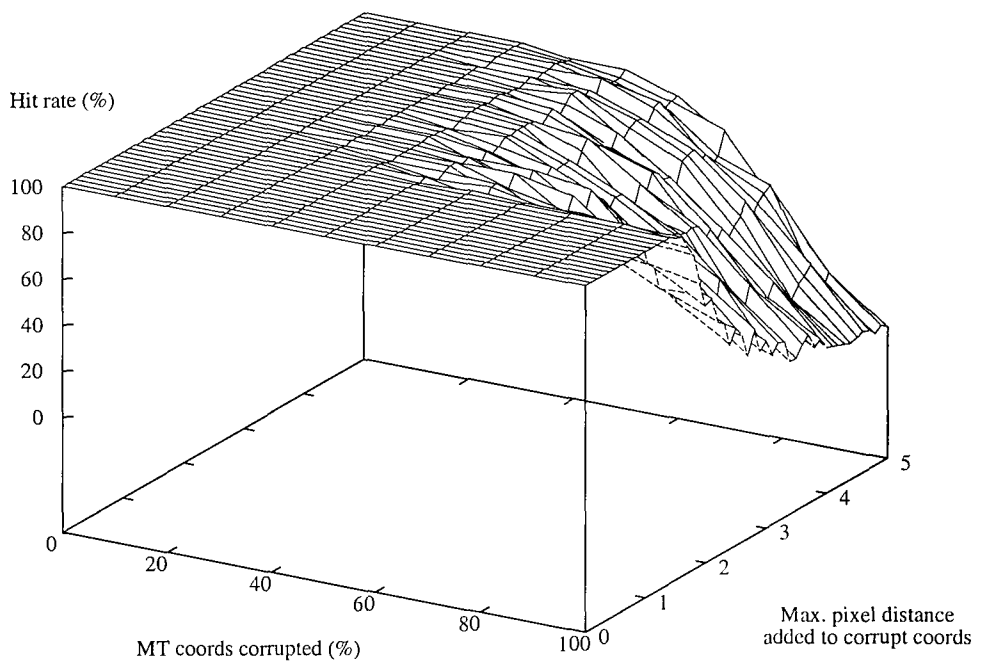


(b) Total misses (inverse of graph above and at a different viewing angle)

Figure 31: Hits and total misses with percentage of corrupted co-ordinates in motion template



(a) Near hits (within one pixel)



(b) Near hits (within three pixels)

Figure 32: Nearby misses with percentage of corrupted co-ordinates in motion template

In real situations, only some portions of the motion template are significant - it is a continuous representation being used with a discrete data set (the frames in a sequence). Consequently, each sequence will only exercise certain co-ordinates in the motion template - those with a time reference matching the time references of the frames in the sequence analysed. The effect of this on the validity of the results as a whole is negligible since noise is applied randomly and uniformly with many trials, thus averaging out any effects. However, since it is entirely possible to have a high corruption rate without necessarily affecting the particular co-ordinates that are vital to a sequence, the graphs show a smooth decline rather than a plunge.

The graphs all show a plateau effect where the "maximum pixel distance" value is below a threshold (one pixel for Figure 31a and Figure 32a, two pixels for Figure 32b). This occurs when the noise is less than the "hit" threshold. It is due to two effects: the discretisation process of accumulation rounds out the errors to within one pixel of accuracy and the post-processing that determines whether a hit has occurred flattens any other errors up to the hit threshold.

In summary, the motion template is sensitive to noise only when the points significant for the sequence being processed are affected by noise, and if their number is sufficient to overwhelm the in-built resilience of the evidence-gathering approach.

4.4.5 Simulated time-lapse imagery

Replicating the earlier CVHT (Section 4.4.1) tests, this section tests motion template performance in conditions that simulate time-lapse video. Previously, it was stated that time-lapse video can be viewed as regular occlusion of the target and, as such, will cause severe problems for techniques that suffer in occlusion. It was found that the CVHT was successful with 50% of the sequence removed (a time-lapse where one frame in two is kept) but failed thereafter. It is likely that this failure was due to the inaccurate modelling of the motion of the target, a factor that is shown to have been eliminated by the motion template technique.

The same CA1 sequence (but with 40 frames, not 20) has been examined with the same time-lapse and noise (Gaussian wrap-around) models. As before, the number of trials were limited by practical concerns and focussed in areas of interest.

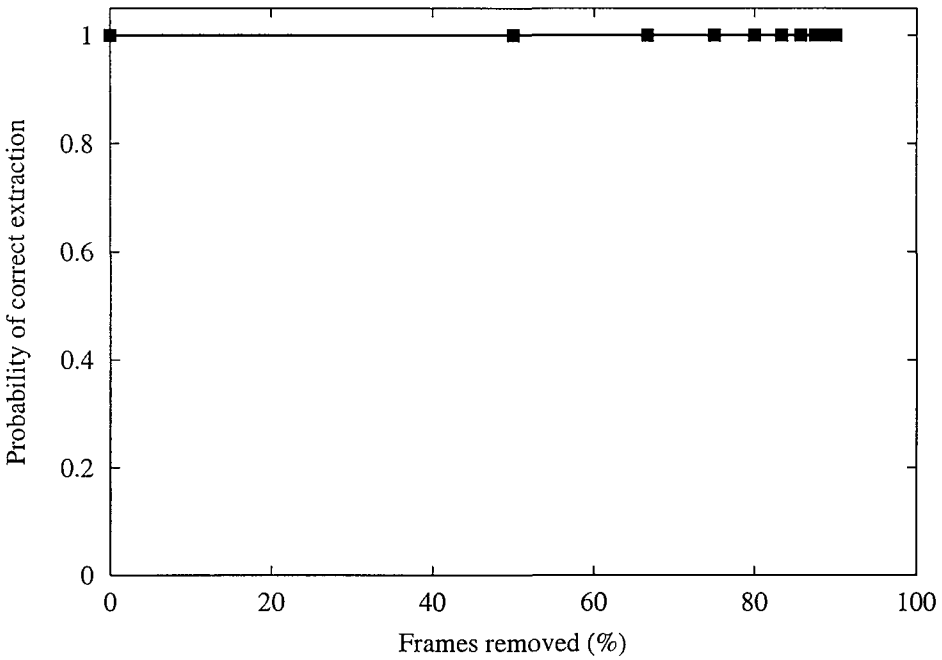


Figure 33: Simulated time-lapse sequence for 40 frames of CA1 sequence

The results in Figure 33 above show that time-lapse does not affect the extraction, even at the highest levels tested (naturally, 100% time-lapse must fail and thus was not included above). The motion template technique models the motion of the target well enough that any one frame is normally sufficient for accurate extraction. Considered another way, when the motion template implementation is given only one frame, it will degenerate to a single frame processor, like the GHT. The only difference is that the motion template algorithm's output will be time-corrected, so that parameters are reported relative to time zero, rather than applying directly to the frame processed as with the GHT. Naturally, the level of trust that can be placed in extractions from very short sequences is limited due to the small amounts of evidence accumulated. However, the motion template technique will always return the best fit for the data provided, even if that quantity of data is insufficient for a trustworthy analysis.

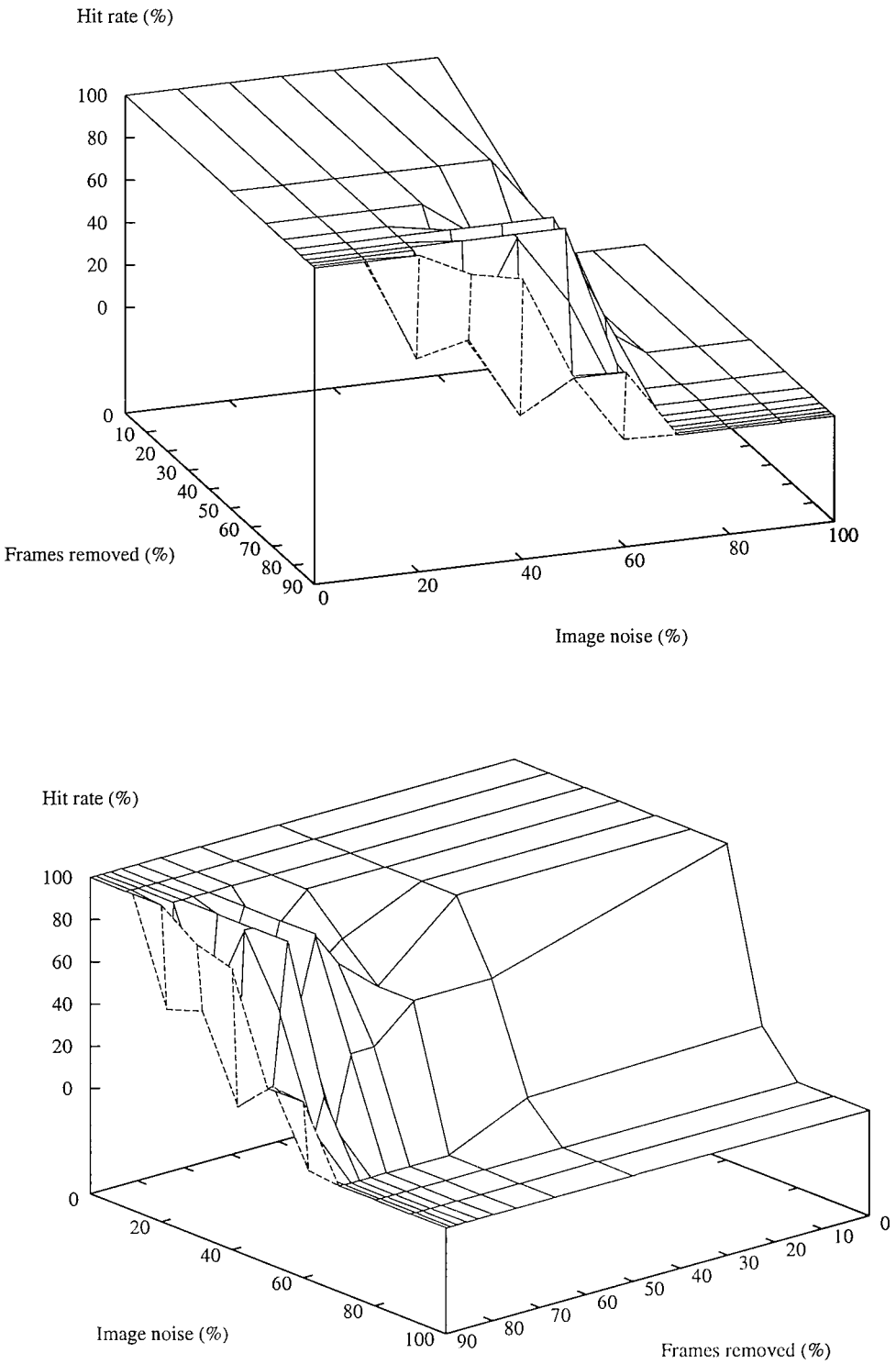


Figure 34: Two views (from different angles) of performance when using simulated time-lapse imagery in varying levels of noise.

A fully successful extraction in all conditions (as above) does not reveal much about the performance of the technique. Consequently, and as with the CVHT tests, image noise is introduced in addition to time-lapse to give a more meaningful performance test. The graphs in Figure 34 indicate that the motion template technique performs appreciably better than the CVHT. Compared to the earlier algorithm (Section 4.4.1), total collapse is reached after 10-20% more image noise is applied (approximately 60-70% image noise), with excellent performance in highly time-lapsed sequences. For the majority of the time-lapse range, the image noise performance curve is notably similar to that displayed in the CA1 real-world tests in Section 4.4.2 above.

Looking at near misses (Figure 35), we see the motion template algorithm has a gently declining period where the output is close to the correct result before errors become prevalent. The period of grace is smaller than that of the CVHT, giving a sharper drop-off. This must be weighed against the fact that the motion template technique is robust in the lower noise levels where the CVHT fails (contrast Figure 35 with Figure 17 in Section 4.4.1). Here, however, noise levels must be significantly higher to produce terminal failure - a flawless hit rate is maintained until contamination by noise is quite excessive (60% image noise with 80% time-lapse). Collapse occurs by 80% image noise at all levels of time-lapse, which is in line with the tests on non-time-lapsed imagery in Section 4.4.1.

These results show the motion template algorithm is significantly more robust than the CVHT, enjoying notable endurance to high levels of time-lapse combined with substantial image noise. The credit for the improved extraction is likely due to the accurate motion model employed, which allows exact extraction on very limited data.

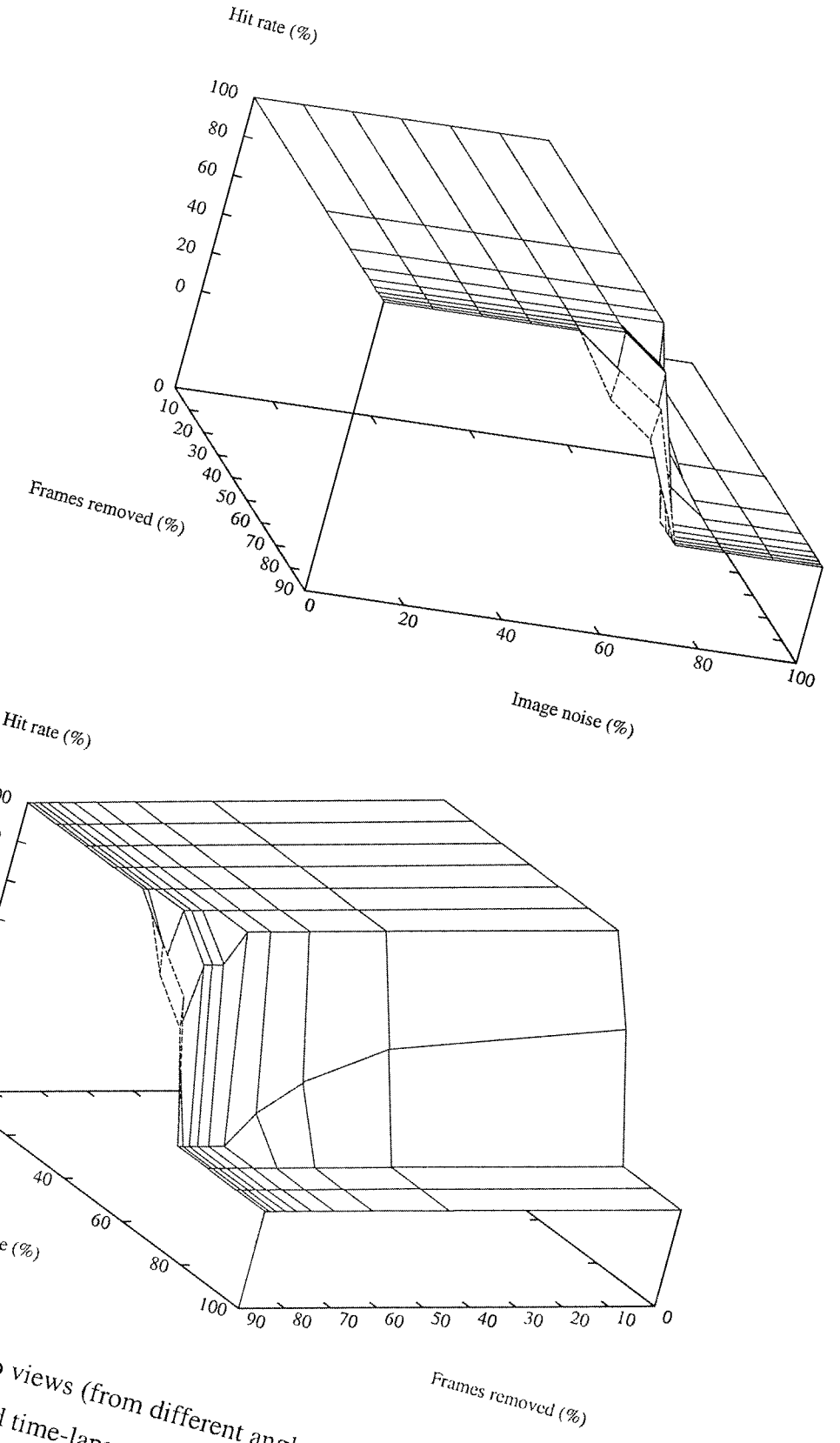
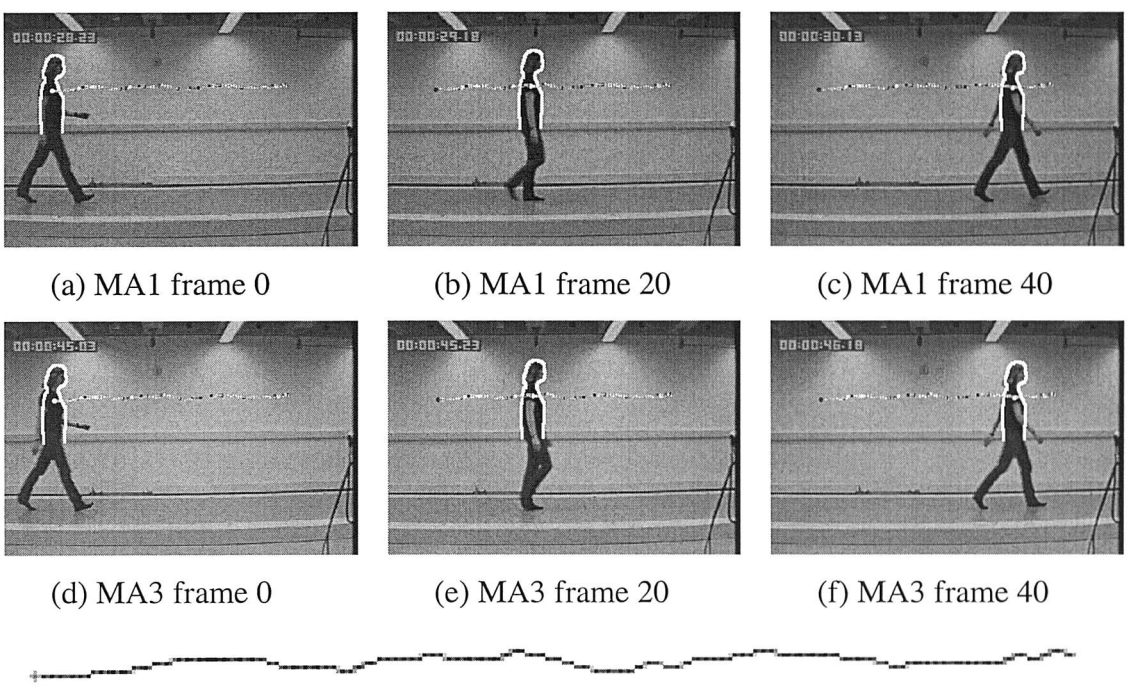


Figure 35: Two views (from different angles) of near miss performance in a simulated time-lapse sequence with varying levels of noise

4.4.6 Finding people with motion templates

Further tests were performed to check the repeatability of human walker extraction. Three extra image sequences of walkers (MA1, SG1 and VH1) were correctly extracted as shown in Figure 36, Figure 37 and Figure 38. Also shown in these figures are the results of attempting to extract the same walker - using the shape and motion templates from the original sequence - but tested on a second sequence (MA3, SG3 and VH3 respectively). This experiment is the beginnings of a study as to whether the shape and motion templates are appreciably similar for a person at different times, or indeed, whether a single generic template can be used for a general walker extraction technique.



(a) MA1 frame 0 (b) MA1 frame 20 (c) MA1 frame 40

(d) MA3 frame 0 (e) MA3 frame 20 (f) MA3 frame 40

(g) Motion template (starts on left)

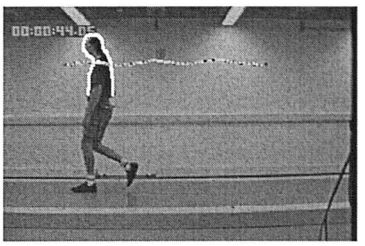
Figure 36: MA1 and MA3 extracted with the same templates

As can be seen, MA3 (Figure 36) has been correctly extracted using the MA1 templates. Using the template from SG1, the extraction of SG3 (Figure 37) begins promisingly but loses accuracy towards the end of the sequence. Examining the full sequence (Appendix 9) shows that the subject moves faster than the motion template predicts in the latter half of the sequence, thus outpacing the template. Close

examination of the second half of the SG1 motion template shows a change from the pattern established in the first half - the subject slows down after he reaches the midpoint of the sequence.

(a) SG1 frame 0

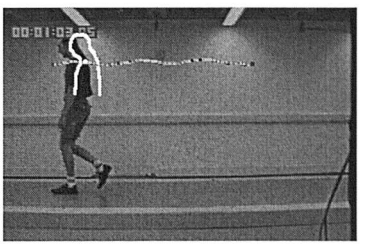
(b) SG1 frame 20



(c) SG1 frame 40

(d) SG3 frame 0

(e) SG3 frame 20



(f) SG3 frame 40

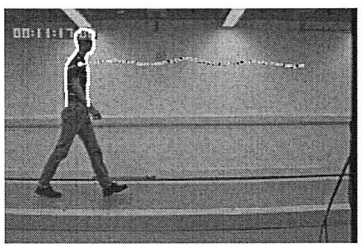
(g) Motion template (starts on right)

Figure 37: SG1 and SG3 extracted with the same templates

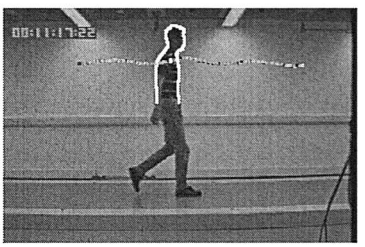
Figure 38 (VH3) demonstrates the same difficulty when using the template for VH1, but the algorithm manifests the location error at the beginning of the sequence. This is due to some problems towards the end of the sequence with poor edges (and hence a corrupted motion template) resulting from height of the subject causing interaction with the lighting in the room.

The algorithm, as expected, performs the best extraction possible given the constraints placed upon it. Even when the motion and shape templates do not accurately match the sequence being examined the algorithm has successfully extracted parameters that best match the problem - locating the walker accurately for as much of the sequence as is possible. Given greater computational facilities (or perhaps the genetic algorithm implementation discussed in the future work - Section 5.2), it would have been interesting to allow the algorithm more flexibility in its matching by permitting it to use a range of scaling factors, etc. Since the results are

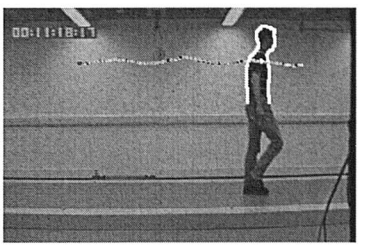
either very accurate or fairly close, this offers hope for a future attempt at locating humans with a "standard walk" motion template and a suitable range of scaling parameters. This has not yet been attempted due to the computational requirements.



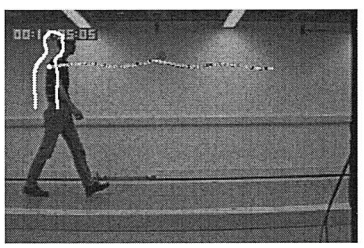
(a) VH1 frame 0



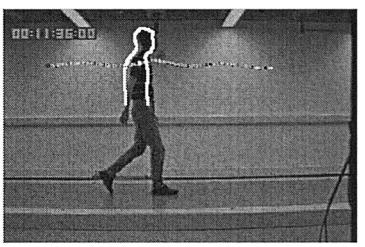
(b) VH1 frame 20



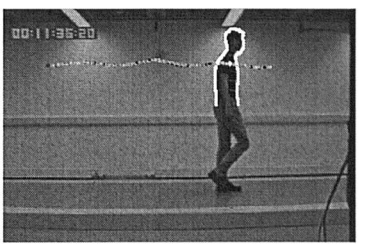
(c) VH1 frame 40



(d) VH3 frame 0



(e) VH3 frame 20



(f) VH3 frame 40

(g) Motion template (starts on left)

Figure 38: VH1 and VH3 extracted with the same templates

Complete image sequences of the subjects studied in this section, with superimposed shape and motion templates, and some example edge-detected images, may be found in Appendix 9.

4.5 Conclusions

This chapter has addressed the issue posed by the CVHT's difficulty with complex motion - namely, that the complexity of the algorithm is proportional to the complexity of the motion model. The central reason for the development of the CVHT was to avoid this very problem, but in the context of shape description rather than motion description. Accordingly, applying the same solution to the motion model was appropriate – i.e. use the template approach to set limits on the complexity.

We have introduced the notion of motion templates as a means for description of a trajectory in order to efficiently integrate arbitrary motion into an evidence-gathering framework. The compromise made is the requirement for prior knowledge of the form of the motion. As argued, the literature has already successfully made a case for shape representation with prior knowledge and the proposition is equally valid for motion representation, given an appropriate application.

Motion templates are a form of temporal template that has been re-interpreted in the HT setting. They have been used to encode multiple components of movement in a fixed cost representation. Thereby, accumulator complexity is constrained to a fixed magnitude regardless of the complexity of the motion. Fourier Descriptors were chosen as the particular form of implementation, although other basis types are equally valid. FDs are especially appropriate in this case since they provide access to the frequency components of the template (a significant factor when examining periodic motion) and, with the CVHT using the same representation for shape, we gain an integrated and consistent framework for the description of both shape and motion. We have provided a theoretical underpinning for this development.

As with the CVHT, we have examined the performance of the new approach in a variety of situations: a set of synthetic circumstances, designed to examine particular characteristics, and a number of real-world test scenarios, in order to ensure the results from synthetic analysis generalise. As before, applying noise (both Gaussian and occlusion) to source images and processing them demonstrated excellent resilience - in fact, the performance was an improvement on the CVHT, necessitating the use of a more vigorous noise function in order to acquire useful results. Similarly, time-lapse studies showed improvements on the CVHT, paralleling those of the noise results. Examination of real-world sequences confirmed that the new algorithm

generalises without significant degradation. It seems likely that the improved performance can be attributed to the extra information possessed by the motion template technique, which facilitates enhanced resolving power in complex scenarios by focussing the peaks in the accumulator.

With the introduction of the motion template, there is a new opportunity for noise to enter the system - noise in the motion template itself. We also examined this source of errors and found performance results consistent with the HT roots of the algorithm. Increasing noise initially has no effect, and then location accuracy suffers from local jitters as the peak in the accumulator begins to spread before finally noise levels cross a threshold and the algorithm fails.

Finally, we have examined the effects of using shape and motion templates on sequences that they were not generated from. This investigation was intended to explore the possibility of using generic templates for particular scenarios and allowing the algorithm to adapt them to the situation. The results showed that the idea has promise; it has excellent accuracy in one case and reasonable success in a further two tests. The limiting factor was the amount of processing required, a solution to which is perhaps a GA, described in the next chapter.

In summary, introducing templates to describe motion gives us a robust, optimal and (within the constraints of an evidence-gathering implementation) efficient algorithm capable of extracting arbitrarily moving arbitrary shapes.

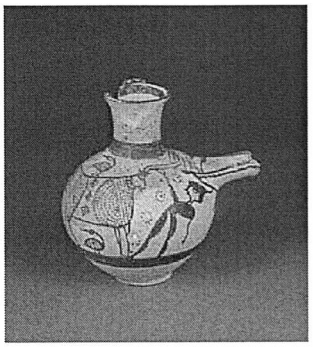
5 Further work

Our suggestions for future development concentrate on two main areas: applications and the algorithm.

In the application domain, we intend to integrate current gait research with the new techniques (see Section 5.1). This is likely to require articulated objects, a simple extension, and the use of optimisation procedures in order to make practical searches on larger parameter spaces with current technology. Consequently, a genetic algorithm version of the motion template technique has been implemented, and is described in Section 5.2.

On the algorithm development side, it would be useful to partly relax the rigid shape requirement in order to further generalise the approach. A possible approach might allow the motion template to specify different models for different parts of the sequence, perhaps with some form of interpolation or morphing between models that is loosely based on the HT for natural shapes (HTNS) [55]. The HTNS uses the set difference (in the mathematical sense) of the two extremes of a shape as the vote pattern, rather than the contour of one shape. In a motion template context, a model is selected for each waypoint in the motion template (e.g. two models may be used, one for waypoint 0 – frame 0 – and one for waypoint 1 – frame 10), and the vote pattern for a particular frame between two waypoints could be derived from an interpolation of the models based on the HTNS principle. Figure 1 illustrates the concept, with Figure 1d showing the proposed accumulator vote pattern for a point in time between the two models in Figure 1b and Figure 1c. The dark areas indicate where votes are cast, with larger areas signifying more uncertainty in the exact location of the true edge. The major disadvantage to this vote pattern is that it assigns more votes (and thus more weight) to areas of uncertainty, therefore increasing the likelihood of distorting the peak.

An alternative to the suggested HTNS development might use a different method to generate the appropriate vote pattern. For example, a traditional computer-graphics "morph" might be used, where the location of particular points on an image are interpolated from the old location to the new, warping the image they are attached to, and whilst the original image is faded out and the new one faded in.



(a) A Mycenaean jug

(b) Vote pattern for jug at 0^{o}

(c) Vote pattern for jug at 15^{o}

(d) Combined vote pattern

Figure 39: Suggested means of representing uncertainty for a changing model

5.1 Gait Recognition

An emerging application of computer vision techniques is that of person identification (e.g. ATM machines using iris recognition, fingerprint or face recognition for unlocking screensavers and doors, see [31] for more examples). Many of the biometric collection mechanisms require close proximity or even contact. They may also be vulnerable to deception (e.g. holding a photograph of someone's face in front of the camera).

Gait is a useful biometric because it can be measured non-invasively, remotely and with no easy means of disguise (since changing or hiding one's walk normally makes one conspicuous). Human motion analysis [2,18] has existed for some time, but there are presently only two means of calculating gait biometrics – statistically and using model based techniques [45].

The first method uses statistical techniques such as moments to measure changes in an image sequence of a walking person and derives a recognisable metric from this. This method does not directly use the motion of walking to aid in the location or identification of the person but rather uses the motion content of the images (or a region of them) as a whole. Some examples of statistical techniques are [27,41, 48].

The second approach uses models to identify regions on the body (e.g. the legs) and to make measurements directly upon these extracted regions (e.g. hip rotation cycles). One particular variant, described by Cunado [13], embodies the VHT to extract the locations of a leg in a sequence and, by analysing the angular motion, automatically generates gait signatures for recognition. By combining magnitude and phase components that were calculated from Fourier analysis of hip rotation, Cunado achieved 100% recognition from a database of 10 subjects – admittedly a database of insufficient size for statistical significance.

The new techniques developed here can fit into either recognition category. For model-based recognition, we intend that they will be used to extract the location and orientation of legs, a gait signature as discussed later. For statistical-based recognition, the approaches isolate the moving human body, as such giving a primer to extract features for statistical based techniques, and thus extending the limited stock of techniques for this purpose.

One possible development branch is to introduce articulation to the CVHT, so that

two legs can be tracked. This would enable a gait signature to be calculated from hip rotation cycles and would improve on the work in [13] since it would be possible to model the legs directly rather than with straight lines. However, the dimensionality of the problem remains high.

A second branch would be to attempt to locate individuals using a motion template that describes a generic walking-pattern, ready for further processing to identify the person in question. Given that the representation of motion templates is inherently suited to describing periodic motion such a template should be easy to create. More ambitiously, it might be possible to look for specific individuals using a motion template to describe a person's unique walking motion. This will require an investigation of whether a motion template of a walking pattern is unique to an individual. Alternatively, in combination with an articulated leg model as described above, motion templates could be used to extract both legs for a more detailed description of the motion.

A further twist to this idea is that since the motion template is implemented using Fourier descriptors, which allow access to frequency information, it may be possible to integrate Cunado's work into the motion template framework. The question to be resolved is whether gait signatures are encoded or can be encoded within motion templates. If this were possible, it would allow for an efficient and exact search of a sequence for a particular individual whose signature is known.

5.2 Genetic Algorithms

Genetic algorithms (GAs) [19] attempt to harness the evolutionary principle to achieve rapid but approximate answers to a problem. The technique improves on the gradient descent search algorithm by introducing an element of random change and a survival of the fittest strategy. Each solution is evaluated for its "fitness" (defined by the problem space and a fitness function) and this metric is used to select out the weaker solutions and combine the stronger. Over many generations or iterations, a population of solutions will tend towards greater, though not necessarily optimal, fitness. The random component (both in the choice of combination of and in mutating the genomes) is intended to prevent populations becoming trapped in local minima by occasionally exploring a totally unconnected part of the search space. This is a little like simulated annealing for gradient descent algorithms but without the "kick" being tied to a specific start point.

In the general case, a genetic algorithm version of any particular HT must encode (in a genome) the parameters that normally describe the accumulator space (e.g. x and y offset, angle and scale factor for a GHT). For examples of the approaches taken see [63] for a GA circle-HT or [56] for a GA GHT-like algorithm. The values of these parameters are then allowed to vary under the action of the GA. The genetic operators – selection, crossover (combination) and mutation – allow prior knowledge to be applied to direct the evolutionary pressure. Given a good understanding of the problem, operators can be crafted that maximise the likelihood of improving a genome by excluding those changes that manifestly will fail or be meaningless. However, even without special purpose operators, GAs provide an effective means of searching a large solution space.

At this point in time, computational power is insufficient for truly large-scale processing with the new algorithms above – the search spaces, while constrained, are still too expansive. Consequently, the way to real-world use of this work now is to apply an optimisation technique like a GA. The robustness and optimality of the newly developed HT techniques will be traded for speed by using a GA since it will no longer be performing an exhaustive search. Instead, a genetic algorithm version of any HT effectively reverts to a template-matching implementation. Particular solutions (e.g. the values of x and y offset) are generated by initialisation or combination operators for each member of the population (a genome) and their fitness evaluated for the search space in question. In this instance, the fitness evaluation is made by calculating the number of matches between the outcome predicted by each genome and the actual data in question – i.e. a template match.

We have made a preliminary implementation of a genetic algorithm variant of the motion template HT described in Chapter 4. A brief examination of its performance has been made by attempting to extract the "legs" shape first seen in Section 4.4.1, and shown again in Figure 40a below. As before, we ran multiple trials at several noise levels (100 trials at 11 noise levels). The noise model used was unchanged– Gaussian noise (wrap-around, with a standard deviation of 1 and a mean of 0) added to each pixel. The probability of noise being added to any one pixel was varied between 0% and 100%, in 10% steps.

(a) Sharp edges (b) Smoothed edges (5 × 5 Gaussian blur)

Figure 40: A frame from the "legs" sequence with sharp and blurred edges

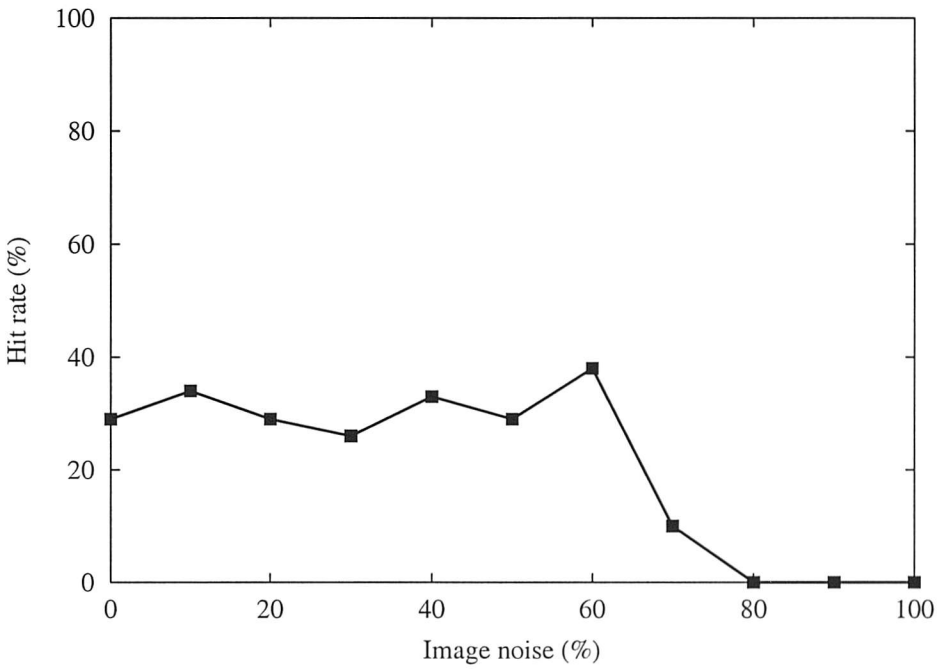


Figure 41: GA performance on sharp-edged sequence in increasing noise (exact hits in 100 trials)

Figure 41 shows the hit rate (defined as an exact match of the location output from the algorithm and the known ground truth). The algorithm is successful in approximately 35%±10% of the trials for the first 7 noise steps (up to 60% noise) before declining to total failure at 80% noise. The shape of this performance curve matches that of the "pure" HT implementation (Section 4.4.1) albeit with a lower

ceiling (~35% versus 100%). This lower ceiling is probably due to the non-optimal nature of genetic algorithms – unfortunate initialisation and getting trapped in local minima will inevitably blight some of the results. Pseudo-random characteristics like initialisation will also account for the fluctuation of the results about the 35% mark. That said, it is worth noting that the GA does far better than chance in attaining the 35% result. Pure chance would give a hit rate of approximately 4% for this test (a small parameter space with a 50 × 50 accumulator and 100 genomes). The graphs below show the hit rates including near misses within one unit by a Euclidean measure (approximately one pixel) - Figure 42 - and three units (approximately three pixels) of the target - Figure 43. They indicate that when the GA misses, it gives a nearby result in 50% of the trials.

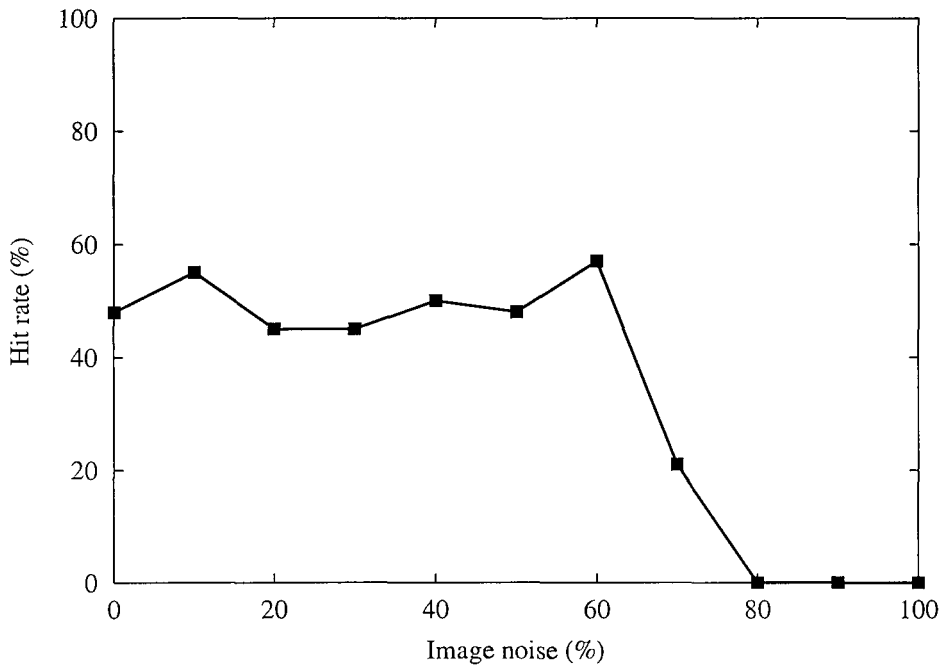


Figure 42: GA performance on sharp-edged sequence in increasing noise - near misses within approx. one pixel of target (includes Figure 41 results)

In comparison with the pure HT implementation, this does not seem impressive. However, the strengths of GAs lie not in their optimality but in their ability to examine a large search space in a practical time. While standard HTs give better results, it is not possible to use them in the more complicated scenarios where the GA compromise remains feasible.

This GA implementation uses standard operators for mutation and crossover. A well-known optimisation is to develop operators specialised for the problem domain. By building in knowledge of the problem, non-beneficial mutations or crossovers can be avoided and the focus of the GA improved. In terms of this algorithm, it may be worth implementing operators that avoid crossover of unrelated parameters. The other area of a GA that has a crucial impact on the solution finding capability is the fitness function. As it stands, the fitness function is merely the result of a template match on the source images for the particular range of parameters encoded in each genome. An alternative function could take account of local curvature (in order to account for spread peaks).

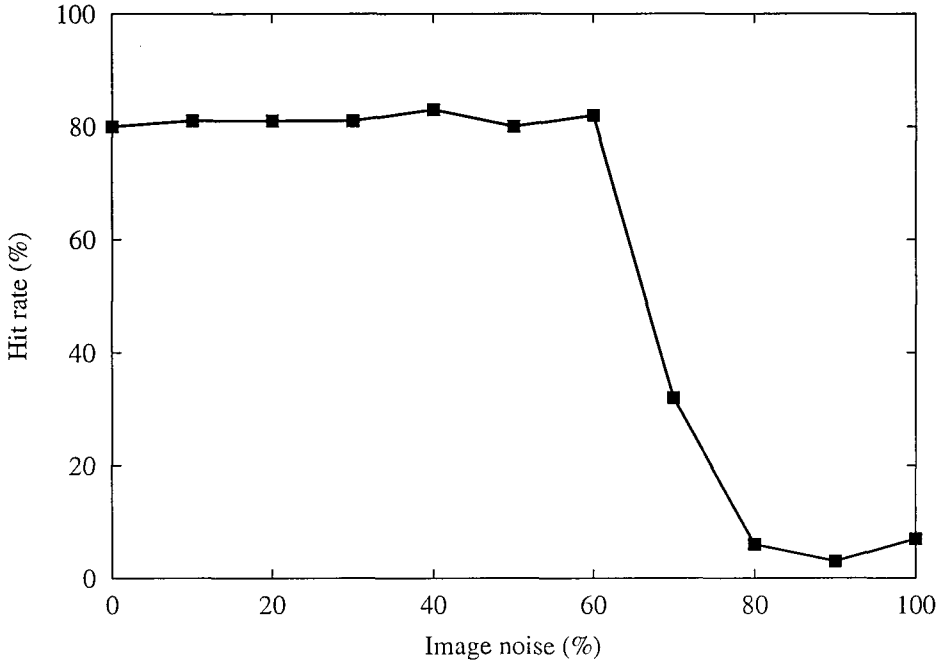


Figure 43: GA performance on sharp-edged sequence in increasing noise - near misses within approx. three pixels of target (includes Figure 42 results)

Cunado [14] has suggested that presenting the GA with a smooth search space aids peak finding by encouraging incremental improvements. He achieved this by using a Sobel edge detector on the source images, thus avoiding the sharp edges of a Canny operator [11]. In the accumulator space (the search space), this is similar to an averaging filter. However, most accumulator spaces show a reasonable degree of continuity due to the nature of the template-matching/HT process; edge pixels in image space contribute to a locus of points in the accumulator, only one of which is

the true peak point. With the loci from many edge pixels converging on a single point in the accumulator, there will inevitably be a "hill" effect from the increasing co-incidences of the loci.

So, the implementation requirements of GAs are very different to those of a HT – as an optimisation procedure, the conditions in which it runs are critical. The nature of GAs implies that they improve incrementally by ascending gradients in the search space to reach maxima. A smooth search space facilitates this ascent by providing "foothills" for the GA to follow up to the main peak. In contrast, the HT is at its best when it has a tightly focussed peak with minimal surrounding vote spread. This conjecture has been briefly examined by applying a Gaussian blur (5 × 5 kernel) to the source images – see Figure 40b for an exemplar image. The blurring of the source images leads to a spreading effect in parameter space, which makes for a smoother ascent for the GA.

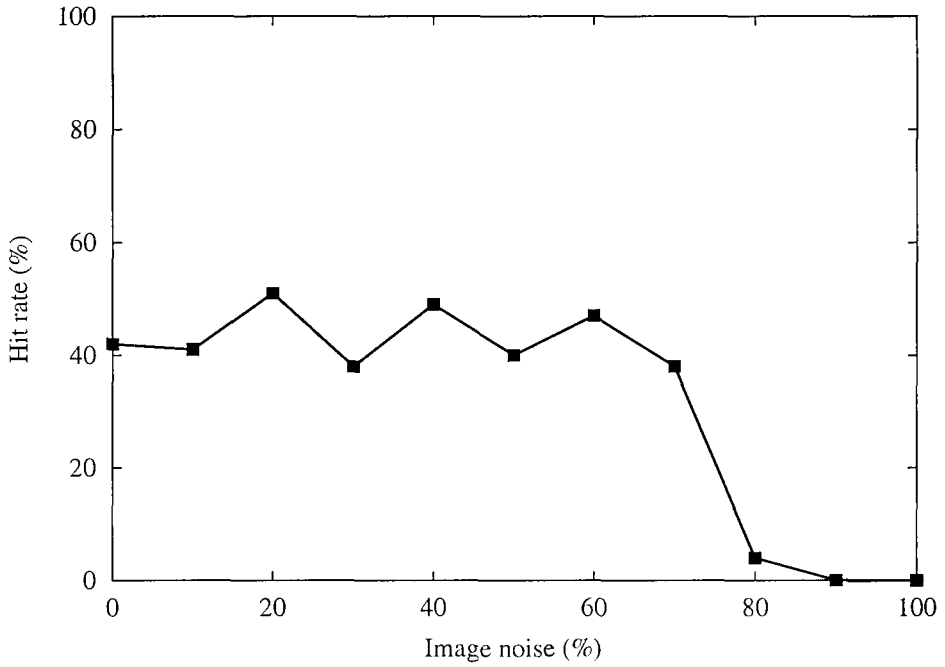


Figure 44: GA performance on smooth-edged sequence in increasing noise (exact hits in 100 trials)

Re-running the trials above gives improved results – see the graphs in Figure 44, Figure 45 and Figure 46. The exact hit performance is approximately 15% higher than before, while near misses within a pixel are approximately 25% higher and near misses within three pixels average 90%, 10% higher than without smoothing.

This brief experiment has shown that GAs, while sub-optimal, are capable of producing the correct result when several trials are made. The performance curve matches that of the pure HT implementation despite operating with a lower peak hit rate. The most significant factor in their favour is that they are a practical way of exploring parameter spaces that are too large for a pure HT implementation.

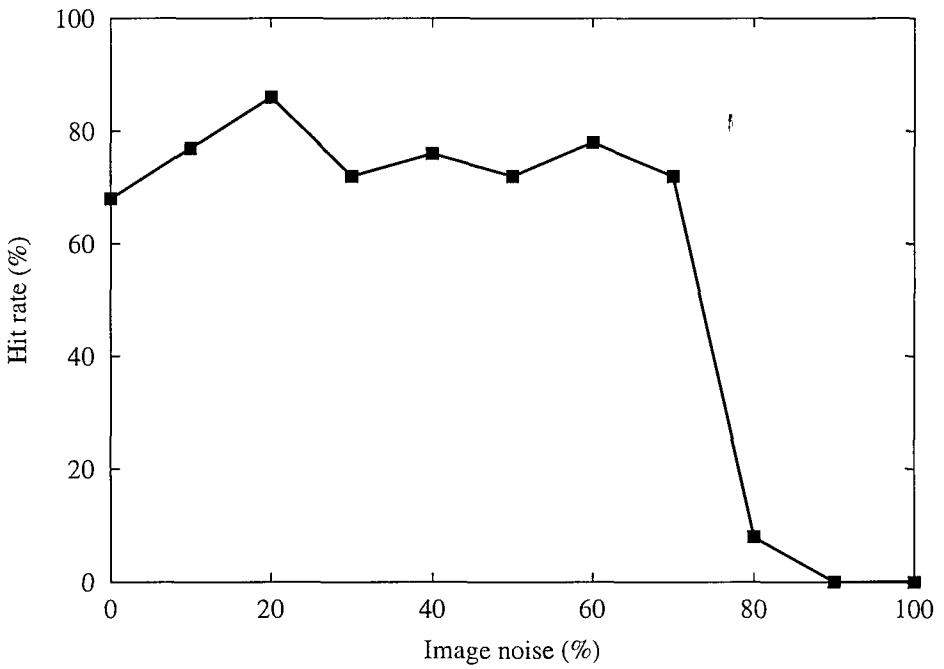


Figure 45: GA performance on smooth-edged sequence in increasing noise - near misses within approx. one pixel of target (includes Figure 44 results)

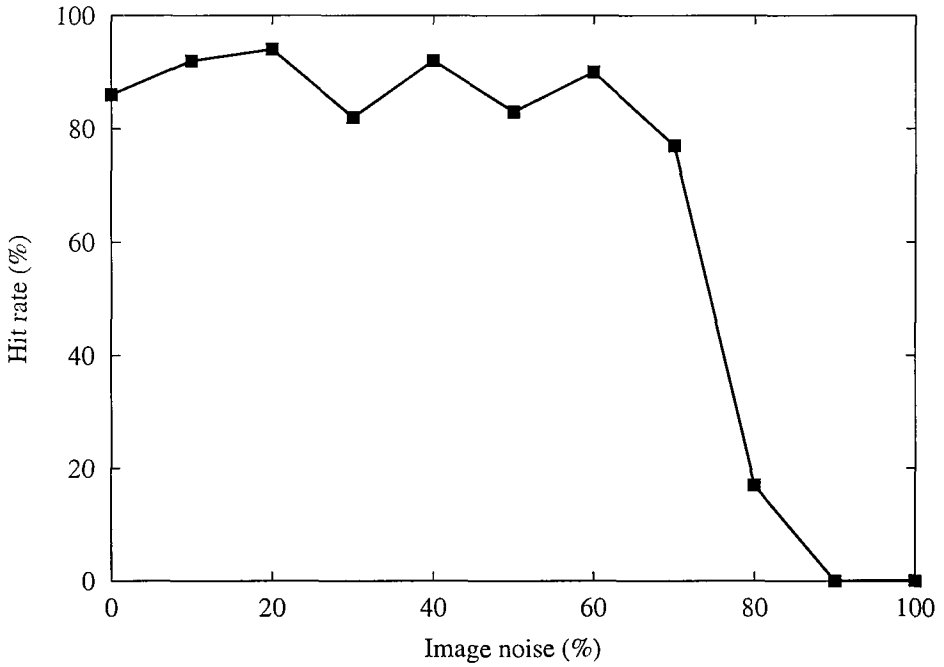


Figure 46: GA performance on smooth-edged sequence in increasing noise - near misses within approx. three pixels of target (includes Figure 45 results)

6 Concluding remarks

6.1 Application scenarios

One of the significant features of this work is the extensive use of *a priori* information regarding target shape and motion. In a number of applications, collecting this information beforehand is infeasible and therefore the algorithms discussed here cannot be used. This bootstrapping problem limits the range of scenarios to which the CVHT and motion template techniques can be applied. However, it has so far proven impossible to produce a totally generic machine vision system, so limitations on applicability are commonplace in the field.

The features of an application that is likely to benefit from the new methods will include:

- Offline, non real-time processing (unless massive computational resources are available),
- Availability of prior knowledge of target behaviour (approximate shape and motion)
- Significant parallelism (ideally)
- Target in a cluttered/noisy environment (take advantage of performance attributes)

An example of an application exhibiting these features is visual database search. The CVHT and motion template algorithms can search, ideally in parallel, a large database of sequences (or a single large sequence that has been split up) that must be examined for known target behaviour. So, for example, if a long sequence is known to have a rare deviation from a norm, and the deviation is predictable in the sense that a human operator can specify it (perhaps by sketching an approximate shape and motion), all occurrences of this deviation can be found. This could be applied to a range of problems – e.g. searching for rare particle tracks in high-energy physics or locating a particular person in a crowd by their shape and motion.

6.2 Implementation issues

6.2.1 Timings

The algorithms described in this thesis were implemented in C++ and run on a variety of machines (most recently a 512MB, 1GHz Pentium-III). Some typical times for a basic run with this hardware setup are listed below:

- CVHT with (x and y) velocity and (x and y) acceleration, running on the Shuttle sequence (27 frames, 240x176), with no noise added. Parameter set in Table 2. Accumulator size = 49,420,800 cells (197,683,200 bytes). Approximately 8,869,899,230 votes cast, taking approximately 7,000 seconds to complete.

Parameter	Number of steps
Image co-ordinate (x)	384
Image co-ordinate (y)	256
Shape scale	1
Shape rotation	1
Velocity in x	1
Velocity in y	1
Acceleration in x	26
Acceleration in y	2

Table 2: Parameter list for CVHT with acceleration

- CVHT with (x only) linear velocity, running on the CA1 sequence (50 frames, 384x256), with no noise added. Parameter set as in Table 3. Accumulator size = 1,179,648 cells (4,718,592 bytes). Approximately 1,052,524,599 votes cast, taking approximately 800 seconds to complete.

Parameter	Number of steps
Image co-ordinate (x)	384
Image co-ordinate (y)	256
Shape scale	1
Shape rotation	1
Velocity in x	1
Velocity in y	1

Table 3: Parameter list for CVHT with linear velocity only

- Motion template HT, running on the CA1 sequence (50 frames, 384x256), with no noise added. Parameter set as in Table 4. Accumulator size = 98,304 cells (393,216 bytes). Approximately 85,526,827 votes cast, taking approximately 900 seconds to complete.
- Motion template HT, running on the CA1 sequence (50 frames, 384x256), with Gaussian wraparound noise (100%) added, using the same parameter set as before (Table 4). Accumulator size = 98,304 cells (393,216 bytes). Approximately 781,028,245 votes cast, taking approximately 8,500 seconds to complete.

Parameter	Number of steps
Image co-ordinate (x)	384
Image co-ordinate (y)	256
Shape scale	1
Shape rotation	1
MT global rotation	1
MT global scale	1
Phase of MT	1
Temporal scaling of MT	1

Table 4: Parameter list for Motion template HT

These timings show a range of times for typical runs. There are two interesting points to note. The first is the factor of ten difference in times between the motion template HT on a clean image and on a 100% noisy image (see Figure 13 for examples). This is due to the high noise causing many more pixels to be active (i.e. not black) and thus requiring that votes be entered into the accumulator for these pixels. It is interesting to note that a simple change in the input image can have such a drastic effect on the time taken to run. This also indicates that, in order to make accurate predictions, the numbers presented here should be used in combination with the complexity notation in Sections 3.1.2 and 4.1.1.

The second point to note is that the motion template HT on the 100% noise sequence takes approximately the same run-time as the CVHT with acceleration, despite the latter casting nearly nine billion votes versus the former's 780 million. This illustrates the impact of the motion template calculations – a fairly minor addition, but one that is compounded over millions of votes.

6.2.2 Optimisations

Some considerable speed increases were achieved by caching important data structures. In particular, the Fourier descriptors require a significant amount of calculation in order to derive a list of discrete co-ordinates representing the analytic shape at a specified scale and orientation. Once these co-ordinates have been calculated, they will be used to cast votes into the accumulator for an edge pixel. After the voting is complete for that pixel, the HT algorithm will move on to the next pixel and exactly repeat the previous process of voting, including requiring exactly the same sets of shape co-ordinates used earlier. Consequently, caching these sets of co-ordinates means that each calculation need only be performed once for each orientation-scale pair. Alternatively, the FD calculation must be made P times, where P is the number of edge pixels in the full image sequence. This gives an order of magnitude improvement, and is especially significant for detailed templates with large numbers of FDs. A quick test confirms this – doing a simple run that took approximately 1,000 seconds with caching was predicted to take approximately 31,000 seconds without (it was aborted after about 4,000 seconds, having processed seven frames out of a total of fifty).

The caching optimization was also deployed for the genetic algorithm code, where the results of expensive template matches (for the specific set of parameters encoded

in each genome) were preserved. Although one might expect this to have little effect because the GA is supposed to be partially searching a large space, actually the mechanism of the GA ensures considerable overlap from one generation to the next. Consequently, many operations can be saved and the performance improvement is noticeable, although difficult to quantify due to the random nature of the algorithm.

The only other important design choice was in the arrangement of the nested loops that are the core of any HT approach. Looking at the pseudo-code in Section 3.3 (CVHT) and Section 4.3 (motion template HT), one will find that the loops are organized so that the most computationally intensive calculations (e.g. generating co-ordinates for shape or motion templates) are in the outermost loops possible. This prevents unnecessary and repeated calculations and keeps the inner loops as clear as possible.

6.2.3 Implementation notes

There are some important issues in the implementation of the Fourier Descriptors that require further detail than given in the main text. In this thesis, we have used Kuhl's Elliptic Fourier Descriptors [35], which are most easily constructed from a chain code. We have used an eight-way chain code, as recommended by Kuhl, which has some accuracy issues.

In terms of our implementation, the chain code algorithm chooses the shortest route around a boundary, which has a tendency to round off right-angle corners. This probably assists the FD encoding later on since there will be fewer high-frequency components resulting from the corners. On the other hand, the implementation is perfectly capable of handling sharp discontinuities to the required accuracy, as proven by its ability to encode the 180° direction change at the ends of a non-cyclic curve (see Section 2.2, final paragraph).

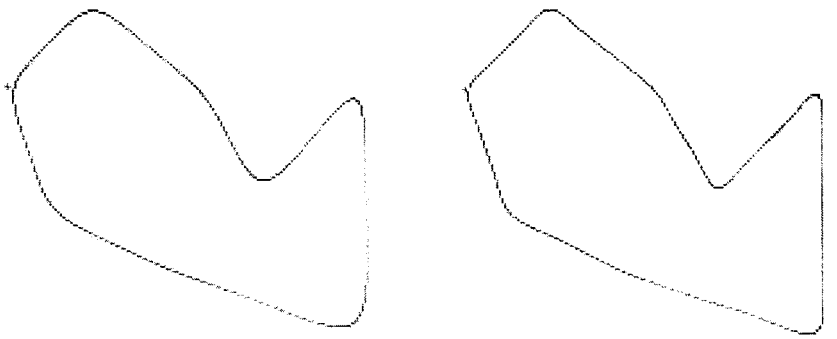
More seriously, there are proven accuracy issues with digital piecewise-linear encoding of continuous curves. Most contour representations incorrectly estimate the total contour length. The representation used here for shapes, Kuhl's Elliptic FDs, uses the (n_e, n_o) characterization, which is shown to have a deviation of 6.6% [16,36]. Interestingly, the adapted "waypoint" representation used for the motion templates is far better in the sense that it accurately specifies the exact length of the segments of the template. With the chain codes, a curve must be built up from many short segments of length 1 or $\sqrt{2}$, whilst the waypoint notation defines the changes from one

waypoint to another with the same precision as the original specification (i.e. a single large linear segment at a shallow angle is encoded as exactly that, rather than a series of steps). As a result of this representation, the motion templates have more accurate contour lengths than the shape templates.

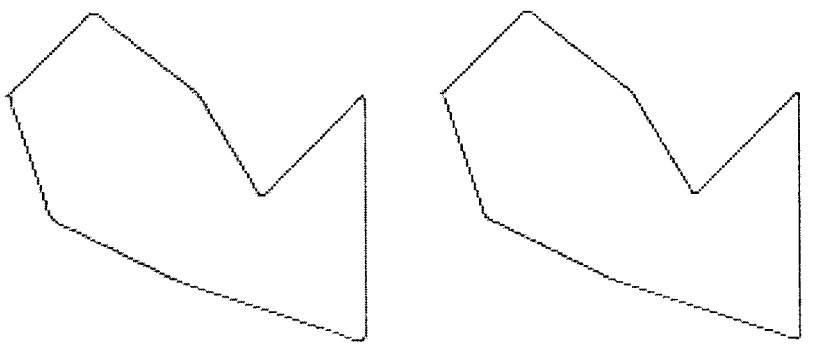
As mentioned in Section 2.2, the accuracy of the representation is dependent on the number of FD harmonics used. Obviously, using too many FDs incurs additional and substantial computational load, whilst using too few gives a bad representation. There is a balance to be struck between asymptotic improvement in accuracy, computational costs and the requirements of the application.

The minimum accuracy we require here is that there must be no sign of aliasing between waypoints at the maximum resolutions requested. An additional factor is that the computational load has been reduced by the use of caching, which makes reconstruction from FDs a fixed cost per sequence and parameter combination. In view of the accuracy requirements and the efficiencies gained by caching, we have generally chosen a constant number of FDs that has proven sufficient at the resolutions and scales chosen. Typically, this figure is 100 FDs for shape templates and simple motion templates. With some of the most complex motion templates, we used 800 FDs – almost certainly overkill, but guaranteed to be enough. Figure 2 shows a motion template reconstructed with various numbers of harmonics, and scaled up by a factor of eight to show detail. Whilst Figure 2a-c show clear evidence of insufficient harmonics (most prominently, rounded corners), Figure 2d and Figure 2e are only marginally different. Figure 2e and Figure 2f are virtually indistinguishable at this resolution, justifying the choice of a constant number of FDs.

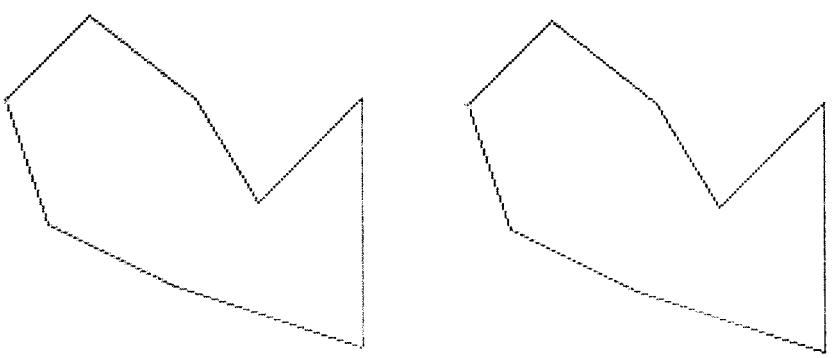
A more automated approach might have used Kuhl's procedure for estimating the maximum error of a curve for a given number of FDs (referred to in Section 2.2). Combine this with knowledge of image resolutions and the sizes of the various scaling parameters, and one could calculate a suitable number of FDs.



a) 8 FDs b) 16 FDs



c) 30 FDs d) 50 FDs



e) 100 FDs f) 1000 FDs

Figure 47: A motion template, reconstructed from FDs with various numbers of harmonics

6.3 Thesis summary and overall conclusions

When the work in this thesis was begun, the state of the art in the HT arena (as applied to moving shape extraction) was to be able to extract analytically moving, analytic shapes using the VHT. Tandem development of a Fourier Descriptor based HT for arbitrary shapes allowed robust extraction whilst minimising discretisation errors.

The first novel development of this thesis was to follow the clear route to increasing the generality of the VHT and FD-variant GHT by combining them. The resulting algorithm, the CVHT, retains the strengths of both its parent algorithms, permitting the optimal extraction of arbitrary shapes that move in a parametric fashion though a sequence of images. The use of FD-based shape templates prevents the rapid expansion of accumulator dimensionality implicit in the previous VHT approach. Comparative tests of the CVHT and a GHT-based frame-by-frame tracking algorithm showed the former is capable of sustaining higher levels of noise than the latter, particularly when individual frames are seriously damaged. The integration of the whole sequence into one accumulator permits exploitation of temporal correlation, thus enhancing the resilience of the CVHT to noise. Experiments on synthetic imagery demonstrated the limits on this resilience and clearly show the importance of temporal correlation in successful extraction. Real world imagery substantiated the results given by the synthetic tests. An alternative motion model was used on a Shuttle launch sequence, simultaneously illustrating the generality of the algorithm and suggesting the next area for study.

The second novel development follows directly from the problem of motion model complexity. The CVHT represents its motion model by the same parametric process that the VHT used for both its shape and motion models. As with the VHT, this process directly links the complexity of the motion model (in terms of the number of its parameters) with the complexity of the algorithm. So, a more complex motion model increases the computational effort in an exponential fashion. To sidestep this issue, we have introduced the motion template to the HT as a way of representing arbitrarily complex trajectories. We used FDs again to obtain the benefits discussed previously and to exploit synergies in a common architecture for model representation, be it shape or motion. Of particular interest is the access to frequency

information afforded by Fourier methods, with possible future application to periodic motion analysis. The new algorithm has been performance tested with synthetic and real-world imagery, using sequences that were occluded, time-lapsed and noisy (both in the image and in the motion template). The results have shown motion templates complement the CVHT, giving more robust extraction than the earlier technique.

Initial research into genetic algorithms to allow the use of wider ranges of parameters on current hardware has shown preliminary success. GAs offer the hope of extracting targets using generic templates that are adapted to the circumstance by the algorithm.

In summary, this thesis has presented a new technique that robustly extracts optimal structural and motion parameters for arbitrarily moving arbitrary shapes in an image sequence. The technique requires no initialisation or training and has demonstrated excellent tolerance to noise and occlusion. Discretisation errors are minimised in the accumulator by using Fourier descriptors to represent templates of both shape and motion in continuous form, which eliminates common problems to do with rotation and scaling. Whilst the templates minimise the effects of noise in algorithm implementation, the temporal correlation between frames is also exploited, maximising the possibility of correct extraction. The use of motion templates is a novel development for the HT and allows for a wide range of applications that require a more general motion model. This new capability comes without the explosion of parameter space dimensionality that would be inherent in current parametric approaches.

7 References

[1] J. K. Aggarwal, Q. Cai, W. Liao and B. Sabata, Nonrigid Motion Analysis: Articulated and Elastic Motion, *Computer Vision and Image Understanding*, **70**(2), pp. 142-156, 1998.

[2] J. K. Aggarwal and Q. Cai, Human Motion Analysis: A Review, *Computer Vision and Image Understanding*, **73**(3), pp. 428-440, 1999.

[3] A. S. Aguado, M. E. Montiel and M. S. Nixon, Improving parameter space decomposition for the Generalised Hough Transform, *Proceedings of the International Conference on Image Processing*, **III**, pp. 627-630, 1996.

[4] A. S. Aguado, M. S. Nixon and M. E. Montiel, Parameterising Arbitrary Shapes via Fourier Descriptors for Evidence-Gathering Detection, *Computer Vision and Image Understanding*, **69**(2), pp. 202-221, 1998.

[5] A. S. Aguado, M. E. Montiel and M. S. Nixon, Bias Error Analysis of the Generalised Hough Transform, *Journal of Mathematical Imaging and Vision*, **12**, pp. 25-42, 2000.

[6] A. S. Aguado and Eugenia Montiel and Mark S. Nixon, *On the Intimate Relationship between the Principle of Duality and the Hough Transform*, Proceedings of the Royal Society A: Mathematics, Physics and Engineering, **456**, pp.503-526, 2000.

[7] J. R. Bennet and J. S. MacDonald, On the Measurement of Curvature in a Quantized Environment, *IEEE Transactions on Computers*, **C-24**(8), 1975.

[8] D. H. Ballard, Generalising the Hough Transform to Find Arbitrary Shapes, *Computer Vision, Graphics and Image Processing*, **13**, pp. 111-122, 1981.

[9] A. F. Bobick and A. D. Wilson, A State-Based Approach to the Representation and Recognition of Gesture, *IEEE Transactions Pattern Analysis and Machine Intelligence*, **19**(12), pp 1325-1337, 1997.

[10] J. E. Boyd and J. J. Little, Global versus structured interpretation of motion: Moving light displays, *Proceedings of the IEEE Computer Society Workshop on Motion of Non-Rigid and Articulated Object, Puerto Rico*, 1997, pp. 18-25.

[11] J. Canny, A Computational Approach to Edge Detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **8**(6), pp. 679-698, 1986.

[12] C. Cedras and M. Shah, Motion-based recognition: a survey, *Image and Vision Computing*, **13**(2), pp. 129-155, 1995.

[13] D. Cunado, J. M. Nash, M. S. Nixon and J. N. Carter, Gait Extraction and Description by Evidence-Gathering, *Proceedings of the 2nd International Conference on Audio- and Video-Based Biometric Person Authentication AVBPA99,* Washington D.C., pp. 43-48, March 1999.

[14] D. Cunado, Automatic Gait Recognition via Model-Based Moving Feature Analysis (*Doctoral thesis, University of Southampton, UK*), 1999.

[15] E. R. Davies, *Machine Vision: Theory, Algorithms, Practicalities*, Academic Press, 1990.

[16] L. Dorst, Length Estimators for Digitized Contours, *Computer Vision, Graphics and Image Processing*, **40**, pp 311-333, 1987.

[17] R. D. Duda and P. E. Hart, Use of the Hough transform to detect lines and curves in pictures, *Communications of the ACM*, **15**(1), pp. 11-15, 1972.

[18] D. M. Gavrila, The Visual Analysis of Human Movement: A Survey, *Computer Vision and Image Understanding*, **73**(1), pp. 82-98, 1999.

[19] D. Goldberg, *Genetic algorithms in search, optimization and machine learning*, Addison-Wesley, 1988.

[20] M. G. Grant, M. S. Nixon and P. H. Lewis, Finding Continuous-Model Moving Shapes by Evidence Gathering. *Proceedings of the IEE Colloquium on Motion Analysis and Tracking*, pp.14/1-14/4, 1999.

[21] M. G. Grant, M. S. Nixon and P. H. Lewis, Finding Moving Shapes by Continuous-Model Evidence Gathering, *Proceedings of the British Machine Vision Conference (BMVC99)*, **2**, pp. 554-563, 1999

[22] M. G. Grant, M. S. Nixon and P. H. Lewis, Parameterised Moving Shape Extraction. *Proceedings of the British Machine Vision Conference (BMVC2000)* pp. 202-211, 2000.

[23] M. G. Grant, M. S. Nixon and P. H. Lewis, Extracting moving shapes by evidence gathering, *Pattern Recognition*, accepted for publication, 2001.

[24] W. E. L. Grimson and D. P. Huttenlocher, On the sensitivity of the Hough Transform for Object Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12**(3), 1990.

[25] J. H. Han, L. T. Koczy, T. Poston, Fuzzy Hough transform, *Pattern Recognition Letters*, **15**(7), pp.649-58, 1994.

[26] P. V. C. Hough, *Method and Means for Recognising Complex Patterns*, US Patent 3069654, 1962.

[27] P. S. Huang, C. J. Harris and M. S. Nixon, Recognizing Humans by Gait via Parametric Canonical Space, *Journal of Artificial Intelligence in Engineering*, **13**, pp. 359-366, 1999.

[28] T. S. Huang and A. N. Netravali, Motion and Structure from Feature Correspondences: A Review, *Proceedings of the IEEE*, **82**(2), pp. 252-267, 1994.

[29] J. Illingworth and J. Kittler, The Adaptive Hough transform, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **9**(5), pp. 690-697, 1987.

[30] J. Illingworth and J. Kittler, A survey of the Hough transform, *Computer Vision and Image Understanding*, **48**, pp. 87-116, 1988.

[31] A. K. Jain, R. Bolle and S. Pankanti Eds: *Biometrics: Personal Identification in Networked Society*, Kluwer Academic Publishers, 1998.

[32] I. A. Kakadiaris and D. Metaxas, Model based estimation of 3D human motion with occlusion based on active multi-viewpoint selection, *Proceedings of the Conference on Computer Vision and Pattern Recognition*, San Francisco, 1996, pp. 81-87

[33] C. Kimme, D. H. Ballard and J. Sklansky, Finding circles by an array of accumulators, *Communications of the ACM*, **18**(2), pp. 120-122, 1975.

[34] N. Kiryati and A. M. Bruckstein, Antialiasing the Hough Transform, *CVGIP: Graphical Models and Image Understanding*, **53**(3), pp. 213-222, 1991.

[35] F. P. Kuhl and C.R. Giardina, Elliptic Fourier Feature of a Closed Contour, *Computer Graphics and Image Processing*, **18**, pp. 236-258, 1982.

[36] Z. Kulpa, Area and perimeter measurement of blobs in discrete binary pictures, *Computer Graphics and Image Processing*, **6**, pp. 434-454, 1977.

[37] W.C.Y. Lam, K. S. Y. Yuen and D. N. K. Leung. Fourier Parameterization provides uniform bounded Hough Transform space, *In:* D. Chetverikov and W. G. Kropatsch, Eds., Lecture Notes in Computer Science **719** (*5th International Conference on Computer Analysis of Images and Patterns '93, Budapest*), Springer Verlag, Berlin/New York, pp. 183-190, 1993.

[38] V. F. Leavers, Which Hough Transform?, *Computer Vision, Graphics and Image Understanding: Image Understanding*, **58**, pp. 250-264, 1993.

[39] P. M. Merlin and D. J. Farber, A Parallel Mechanism for Detecting Curves in Pictures, *IEEE Transactions on Computers*, **24**, pp. 96-98, 1975.

[40] T. B. Moeslund and E. Granum, A Survey of Computer Vision-based Human Motion Capture, *Computer Vision and Image Understanding*, **81**, pp. 231-268, 2001.

[41] H. Murase and R. Sakai, Moving object recognition in eigenspace representation: gait analysis and lip reading, *Pattern Recognition Letters*, **17**, pp. 155-162, 1996.

[42] M. P. Murray, Gait as a total pattern of movement, *American Journal of Physical Medicine*, **46**(1), pp. 290-332 1967.

[43] J. M. Nash, J. N. Carter and M. S. Nixon, Dynamic Feature Extraction via the Velocity Hough Transform, *Pattern Recognition Letters*, **18**(10), pp. 1035-1047, 1997.

[44] J. M. Nash, J. N. Carter and M. S. Nixon, Extracting moving articulated objects by evidence gathering, *Proceedings of the British Machine Vision Conference (BMVC98)*, **2**, pp. 609-618, Southampton, Sept. 1998.

[45] M. S. Nixon, J. N. Carter, D. Cunado, P. S. Huang, S. V. Stevenage, Automatic Gait Recognition, *In*: A. K. Jain, R. Bolle and S. Pankanti Eds: *Biometrics: Personal Identification in Networked Society*, pp. 231-250, 1998.

[46] M. S. Nixon, *Feature Extraction and Image Processing*, Butterworth-Heinmann, to be published 2001.

[47] W. Niblack and D. Petkovic, On improving the accuracy of the Hough transform: Theory, simulations and experiments, *Proceedings of the IEEE Computing Society*

Conference on Computer Vision and Pattern Recognition, Ann-Arbor, June 1988, pp. 574-579, 1988.

[48] S. A. Niyogi, E. H. Adelson, Analyzing and recognizing walking figures in XYT, *Proceedings of the Conference on Computer Vision and Pattern Recognition 1994*, pp. 469-474, 1994.

[49] N. Peterfreund, The Velocity Snake: Deformable Contour for Tracking in Spatio-Velocity Space, *Computer Vision and Image Understanding*, **73**(3), pp. 346-356, 1999.

[50] R. Polana and R. C. Nelson, Detection and Recognition of Periodic, Nonrigid Motion, *International Journal of Computer Vision*, **23(3)**, pp. 261-282, 1997.

[51] J. Princen, J. Illingworth and J. Kittler, A hierarchical approach to line extraction based on the Hough Transform, *Computer Vision and Image Understanding*, **52**, pp. 57-77, 1990.

[52] J. Princen, J. Illingworth and J. Kittler, A Formal Definition of the Hough Transform: Properties and Relationships, *Journal of Mathematical Imaging and Vision*, **1**, pp. 153-168, 1992.

[53] A. Rosenfeld, *Picture Processing by Computer*, Academic Press, London UK, 1969.

[54] P. Rosin and S. Venkatesh, Extracting natural scales using Fourier descriptors, *Pattern Recognition*, **26**(9), pp. 1383-1393, 1993.

[55] A. Samal and J. Edwards, Generalized Hough transform for natural shapes, *Pattern Recognition Letters*, **18**, pp. 473-480, 1997.

[56] P. K. Ser, C. S. T. Choy and W. C. Siu, Genetic Algorithm for the Extraction of Nonanalytic Objects from Multiple Dimensional Parameter Space, *Computer Vision and Image Understanding*, **73**(1), pp. 1-13, 1999.

[57] G. C. Stockman and A. K. Agrawala, Equivalence of Hough Curve Detection to Template Matching, *Communications of the Association for Computing Machinery*, **20**, pp. 820-822, 1977.

[58] S. Tsuji and F. Matsumoto, Detection of ellipses by a modified Hough Transform, *IEEE Transactions on Computers* **27**, pp. 923-926, 1978.

[59] T. M. Van Veen and F. C. A. Groen, Discretizaton errors in the Hough transform, Pattern Recognition, **14**, pp. 137-145, 1981.

[60] L. Xu, E. Oja and P. Kultanen, A new curve detection method: Randomised Hough Transform (RHT), *Pattern Recognition Letters*, **11**, pp. 331-338, 1990.

[61] Li-Qun Xu and D. C. Hogg, Neural Networks in human motion tracking - An experimental study, *Image and Vision Computing*, **15**, pp. 607-615, 1997.

[62] S. B. Yacoub and J. M. Jolin, Hierarchical line extraction, *IEE Proceedings on Vision, Image and Signal Processing*, **142**(1), pp. 7-14, 1995.

[63] P-Y. Yin, A new circle/ellipse detector using genetic algorithms, *Pattern Recognition Letters*, **20**, pp. 731-740, 1999.

[64] Moving Picture Experts Group (MPEG), WWW home page: http://www.cselt.it/mpeg/

8 Appendix: Formal theory of GHT

This appendix reprises the work in [4] defining formal theory for the GHT. Also noteworthy is work by the same author on the principle of duality [6].

8.1 The Generalised Hough Transform

The first formal definition of the HT was developed by Princen *et al* [52]. Its value is that it facilitates a proper analysis of the behaviour of the HT by providing a yardstick against which performance can be measured and a means of predicting the capabilities of an algorithm. However, their definition is limited in that it only describes parameterised versions of the HT – i.e. only those that define the features analytically (lines, circles, ellipses, etc).

In the formal definition the value at a point in parameter space is represented as:

$$H(\Omega) = \int p(X, \Omega) I(X) dX \tag{20}$$

where $p(X, \Omega)$ is the HT kernel and $I(X)$ represents the feature space (e.g. the image to be searched). The equation is given as an integral since both the parameter and feature spaces are continuous. The HT kernel can be interpreted in two ways: for a fixed value of x, the kernel describes the points in feature space that make up the perfect instance of the parameters (Ω); for a fixed value of Ω, the kernel describes the point-spread function in parameter space (i.e. how the votes are cast for a particular feature point).

$I(X)$ is composed of Dirac delta functions in a continuous space, with each delta function representing one feature point. Using the sifting property of the delta function, the integral can be reduced to a summation to give a more familiar result:

$$H(\Omega) = \sum_{j=1}^{n} p(X_j, \Omega) \tag{21}$$

where n is the number of feature points to be considered. This summation represents the votes cast in parameter space for a particular feature point. The final HT is the summation of $H(\Omega)$ for all Ω, resulting in a double summation that describes the implementation of a typical HT accumulator.

The kernel of the HT is determined by the parameterisation used and the shape of the cells sampling the accumulator. Princen *et al* determined that different cell shapes and parameterisations have an effect on the accuracy of the voting process and can reduce quantisation problems caused by the sampling of the feature space and the parameter space that is necessary for implementation.

The extension of the formal definition to arbitrary shapes [4] has been summarised below to provide context for parts of this thesis. The augmented definition covers both analytically and non-analytically defined shapes. In essence, the extension takes account of the fact that the GHT is the same in concept as that of the HT – the only difference is that an R-table replaces the parameterised equation of a curve (e.g. a circle, line, ellipse, etc). Accordingly, replacing the HT kernel with a description of an R-table arrives at the formal definition of the GHT. With the Merlin-Farber method, the kernel takes the form of a discrete set of points representing the template. The GHT improves on the Merlin-Farber method by reducing the number of feature points considered with a gradient direction constraint. Therefore the kernel appears with a reduced set of discrete points, constrained by the first derivative at each point.

8.1.1 Definition of the HT for non-analytic shapes (general form)

Firstly, a primitive in image space is defined as the set of points in a continuous curve $\bar{z}(s, \bar{a})$, where $\bar{a}$ is a parameter vector that describes the form of the curve and s is a parameter that specifies a point on the curve. This more general formulation is used instead of the implicit form of a curve because it does not restrict the formalism to a specific type of curve. In terms of implementation, Z represents the template in the GHT, where:

$$Z = \{\bar{z}(s, \bar{a}) \mid s \in D_s\} \tag{22}$$

A single point on Z, where $s = s_0$, maps to a curve A_{s_0} in parameter space. A_{s_0} represents the parameters of all the curves that can pass through $\bar{z}(s_0, \bar{a})$. If the function $\bar{z}(s, \bar{a})$ is invertible then $\bar{z}^{-1}(s, \bar{z})$ represents a point in parameter space for a given curve in image space.

$$A_{s_0} = \{\bar{z}^{-1}(s, \bar{z}(s_0, \bar{a})) \mid s \in D_s\}, s_0 \in D_s \tag{23}$$

This is the kernel of the transform, which describes the pattern of votes to be cast in parameter space for a specific point in image space – in this case A_{s_0}. In the HT, A_{s_0} is defined for a set of edge points in image space *I*.

If the image space only contains an instance of a shape defined by a parametric equation $\bar{\lambda}(t)$, then:

$$I = \{\bar{\lambda}(t) \mid t \in D_I\} \tag{24}$$

and A_{s_0} can be redefined as:

$$A_t = \{\bar{z}^{-1}(s, \bar{\lambda}(t)) \mid s \in D_s\} t \in D_I \tag{25}$$

A_t is the set of points in parameter space that represent the possible parameter values for all of the individual edge points in image space *I*. For a specific instance of a shape, Z, only one of the points in A_t will be the correct parameter vector, $\bar{a}$. If the image contains an instance of Z, then all the edge points in that instance will generate curves in parameter space that pass through the point $\bar{a}$. Hence, $\bar{a}$ corresponds to the intersection of the loci of A_t, for all the edge pixels in the image.

$$\bar{a} = \bigcap_{t \in D_I} A_t \tag{26}$$

In the case of noise in image space (ie. edge pixels that do not belong to the primitive Z), loci generated by the noise will generally not pass through $\bar{a}$ and will produce conflicting intersections elsewhere in parameter space. As discussed in more detail below, the HT technique relies on the correlation of good data outweighing the random effects of noise in parameter space.

The equations above describe the concept of the HT but do not formalise the actual technique used, namely the accumulation phase. The parameter space can be mapped into an accumulator by using a matching function, which determines whether a point, $\bar{b}$, in parameter space should be incremented for a point, $\bar{d}$, in the set A_t. The equation below defines the simplest accumulation strategy, that of incrementing by unity for each match. Changing the voting functional, *M*, can accommodate more complex strategies.

$$M(\bar{b}, \bar{d}) = \begin{cases} 1 & \text{if } \bar{b} = \bar{d} \\ 0 & \text{if } \bar{b} \neq \bar{d} \end{cases} \tag{27}$$

Using this function, the HT of the function $\bar{\lambda}(t)$ is defined as:

$$S_{HT}(\bar{b}) = \iint M(\bar{b}, \bar{z}^{-1}(s, \bar{\lambda}(t)))\, ds\, dt \tag{28}$$

The maximum of this function is $\bar{a}$, the parameters best matching Z.

In practise the HT is discrete, not continuous, being quantised by the digitised image and accumulator. A representation of this can be obtained by converting the integrals to summations and changing the continuous function $\bar{\lambda}(t)$ to a discrete set of points $\bar{\lambda}_t$:

$$S_{DHT}(\bar{b}) = \sum_{t \in D_t} \sum_{s \in D_s} M(\bar{b}, \bar{z}^{-1}(s, \bar{\lambda}_t)) \tag{29}$$

8.1.2 Definition of the HT for non-analytic shapes (Merlin-Farber)

The equations above deal with a general formalism of the HT for arbitrary shapes. However, both the Merlin-Farber algorithm and the GHT algorithm can be described more specifically.

For each edge pixel in the input image, the Merlin-Farber algorithm draws the template shape in parameter space, rotated by 180 degrees and with a chosen reference point centred on the co-ordinates of the edge pixel.

First then, the representation of the primitive Z must be adapted to reflect the method used. The template is a set of points $W = \{\bar{w}_B \mid B \in D_B\}$ defined relative to an arbitrarily chosen reference point $\bar{R}$. The primitive can be expressed as:

$$Z = \{\bar{z}_B \mid \bar{z}_B = \bar{R} + \bar{w}_B, B \in D_B\} \tag{30}$$

With this definition, the parameter vector (cf. $\bar{a}$) to be calculated is $\bar{R}$, the location of the reference point. Consequently, the kernel of the HT is defined by:

$$R_t = \{\bar{\lambda}_t - \bar{w}_B \mid B \in D_B\}, t \in D_I \tag{31}$$

In the equation above, the template points $\bar{W}_B$ are translated to the image edge point $\bar{\lambda}_t$ and reflected in the diagonal axis ($y = -x$), i.e. subtracting $\bar{W}_B$ from $\bar{\lambda}_t$. With this kernel, the HT accumulates evidence of a rotated template, translated to the co-ordinates of each edge pixel, as follows:

$$S_{NA}(\bar{b}) = \sum_{t \in D_t} \sum_{B \in D_B} M(\bar{b}, \bar{\lambda}_t - \bar{w}_B) \tag{32}$$

In this discrete form, it can be seen that parameter space, image space and the template must share the same quantisation (i.e. cell size and shape) for the accumulation process to work correctly. If the points in the template are of a different size or shape to the cells in the accumulator, then drawing traces of the template in the

accumulator is meaningless. This is a particular problem with rotated or scaled templates because a rotated/scaled discrete template is effectively quantised in a different way to the original image and accumulator. For example, a rotated discrete template has a different cell shape because the rotation causes the square pixels to change shape relative to the original (e.g. 45 degree rotation will give diamond shaped cells). Similarly, scaling adjusts the cell size. Re-sampling the template into the correct quantisation allows the template to be usable but this process loses information since an alias may be extracted.

8.1.3 Definition of the HT for non-analytic shapes (GHT)

The GHT improves on the Merlin-Farber technique by incorporating gradient direction, reducing the number of votes cast by using the extra information to filter out obviously incorrect votes. In terms of the above, the template W is constrained by the first derivative of the edge pixels in the image I when voting in parameter space. For a vote to be cast, the gradient direction of an edge pixel in the template must be equivalent to that of the image edge pixel being considered. Effectively, the template is still traced out in parameter space but various implausible parts of it are masked out. Therefore, each edge pixel in the search image will generate only a few votes in the accumulator. Fewer votes mean less time is spent in the accumulation stage and less false votes will be cast, reducing the noise in the accumulator.

In terms of the formalisation, this is achieved by considering a function, $\varphi(P)$, which returns the gradient direction G at a point P. If this function is only applied to edge pixels in the template then, when it is inverted, it becomes a function $\varphi^{-1}(G)$ that returns the edge pixels in the template that match its gradient direction parameter. This is the theoretical analogue of the R-table. Using $\varphi^{-1}(G)$, the kernel of the HT becomes:

$$R_t = \left\{\overline{\lambda_t} - \varphi^{-1}(\lambda_t')\right\}, t \in D_t \tag{33}$$

The accumulation process of the GHT is then defined by:

$$S_{GHT}(\overline{b}) = \sum_{t \in D_t} M\left(\overline{b}, \overline{\lambda_t} - \varphi^{-1}(\lambda_t')\right) \tag{34}$$

The second summation that is present in the earlier equation for the Merlin-Farber algorithm voted for all of the points in the template, but here it is unnecessary because $\varphi^{-1}(G)$ returns all the relevant points of the template and thus replaces the second

summation. An alternative definition of the template can restore the second summation:

$$W_t = \{\overline{w}_B \mid \varphi(\overline{w}_B) = \lambda_t', B \in D_B\} \tag{35}$$

Here the template is defined with the gradient direction constraint "built in". The kernel of the HT and the equation for the accumulation process are then nearly the same as that of the Merlin-Farber method.

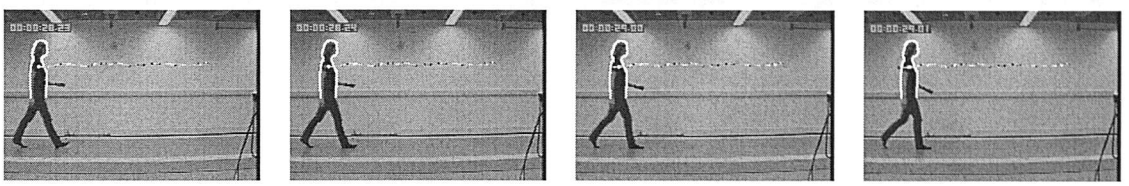
$$R_t = \{\overline{\lambda}_t - \overline{w}_B \mid \overline{w}_B \in W_t\} \tag{36}$$

$$S_{GHT}(\overline{b}) = \sum_{t \in D_I} \sum_{w_B \in W_t} M(\overline{b}, \overline{\lambda}_t - \overline{w}_B) \tag{37}$$

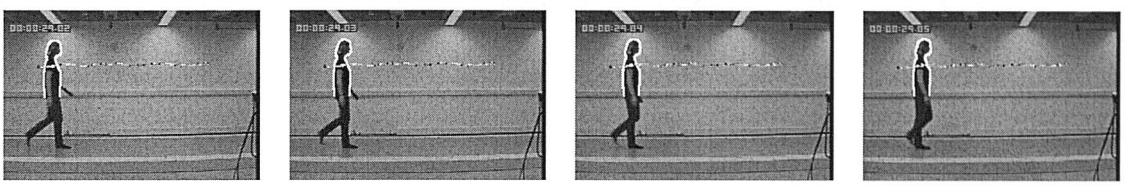
The difference is that the equations for both the kernel and the accumulation process do not consider the template set W to be constant, but rather that it is re-evaluated for each edge pixel in the image being searched.

9 Appendix: Full sequences of motion template person extraction

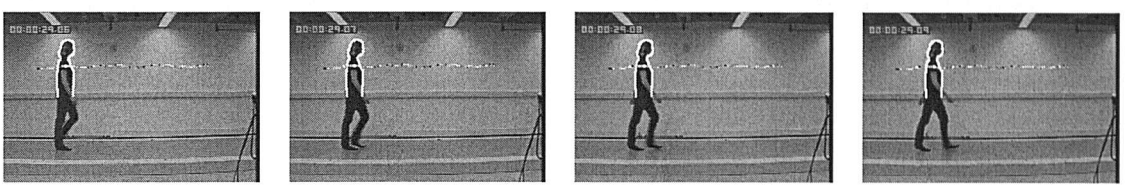
9.1 MA1 extraction with MA1 templates



Frame 0 Frame 1 Frame 2 Frame 3

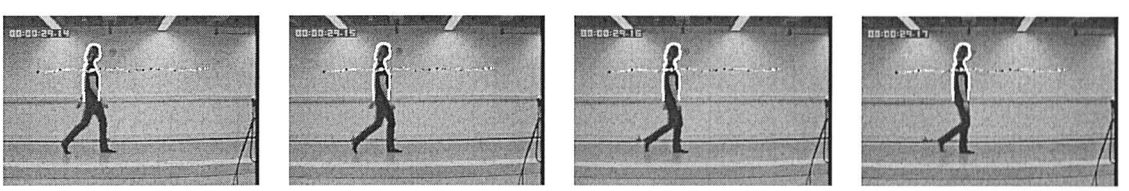


Frame 4 Frame 5 Frame 6 Frame 7

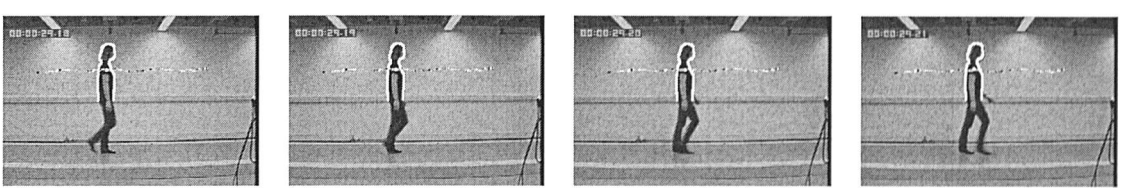


Frame 8 Frame 9 Frame 10 Frame 11

Frame 12 Frame 13 Frame 14 Frame 15



Frame 16 Frame 17 Frame 18 Frame 19



Frame 20 Frame 21 Frame 22 Frame 23

Figure 48: (part 1 of 2) MA1 extracted with the MA1 templates

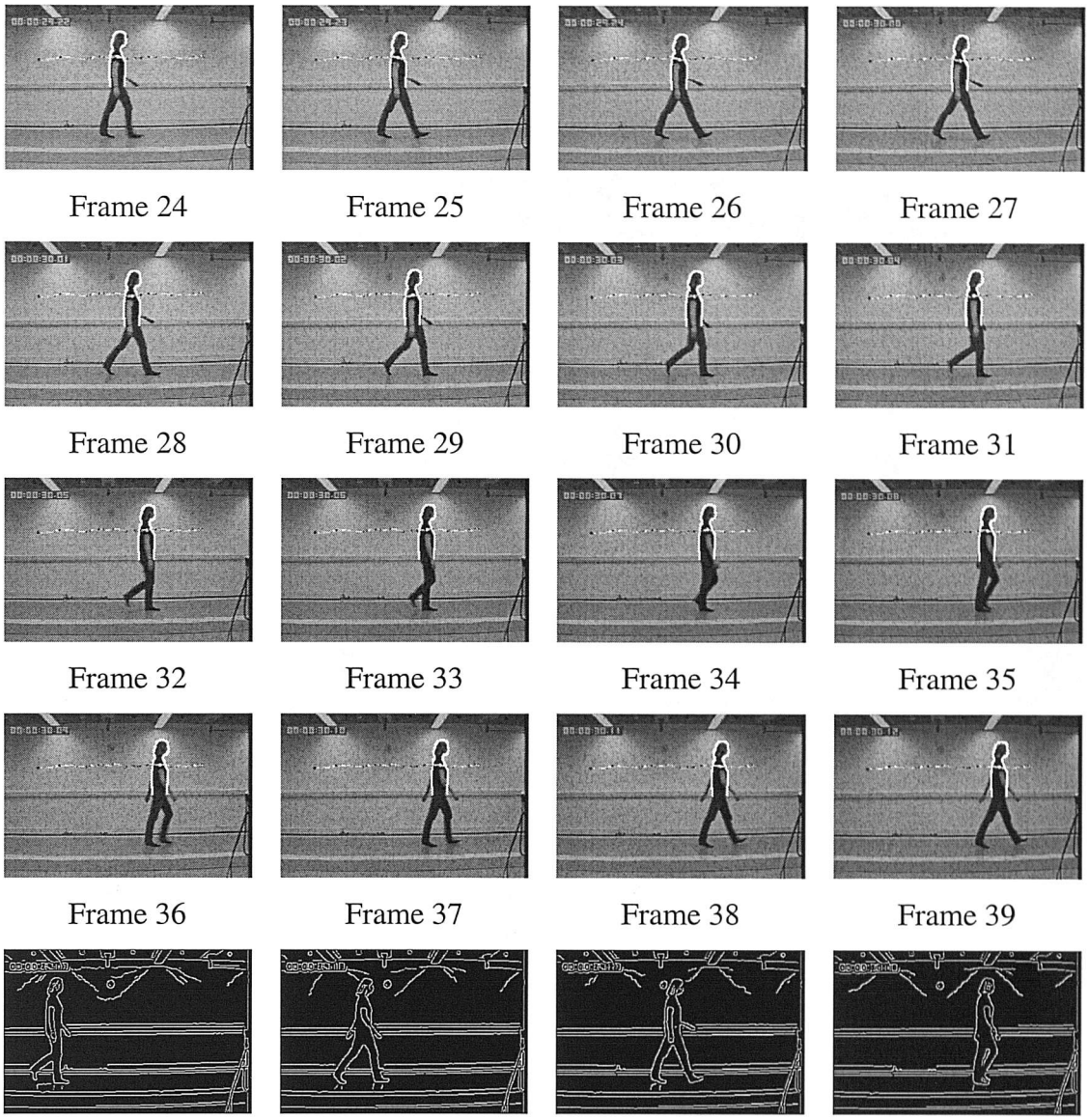


Figure 49: (part 2 of 2) MA1 extracted with the MA1 templates

9.2 MA3 extraction with MA1 templates

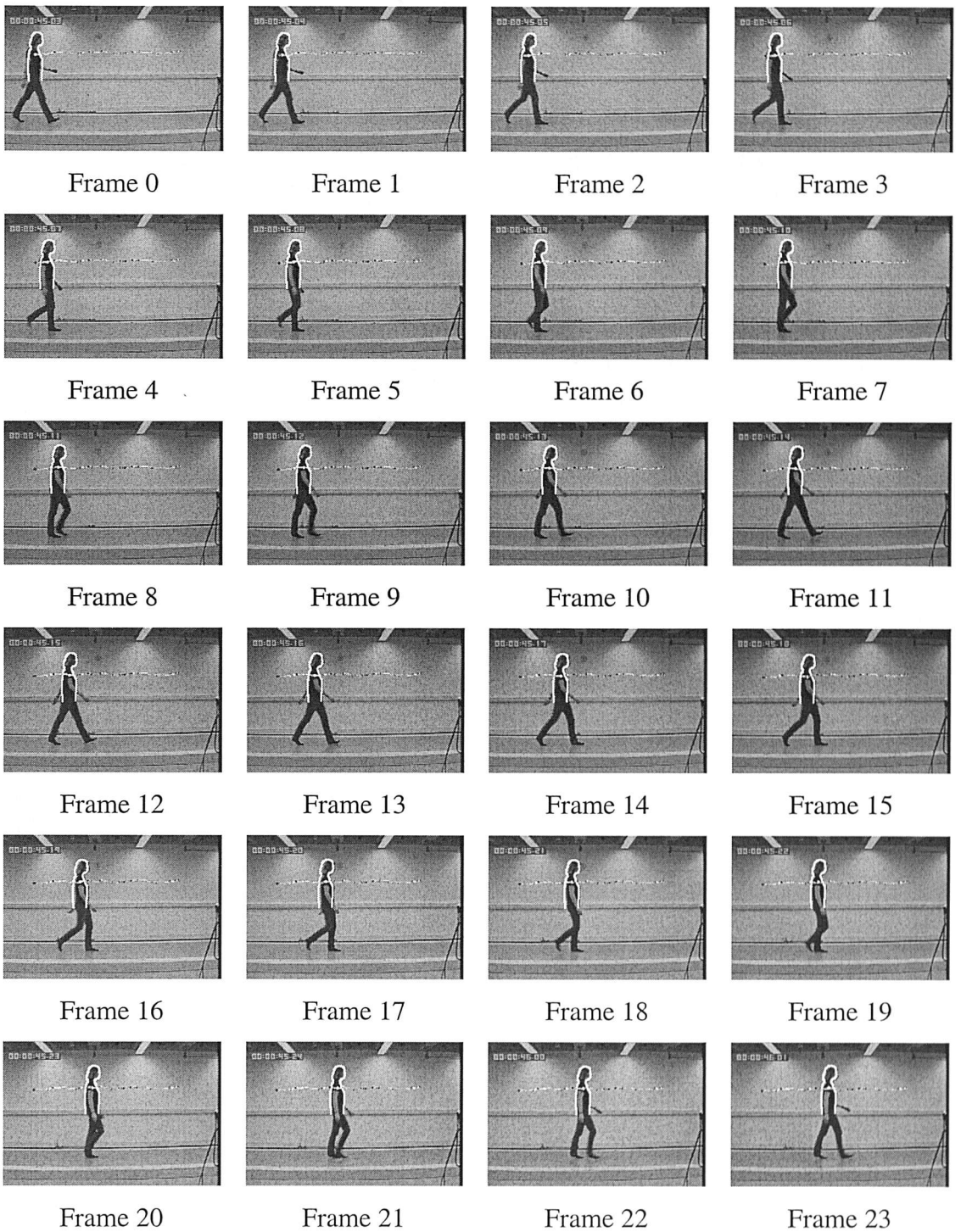
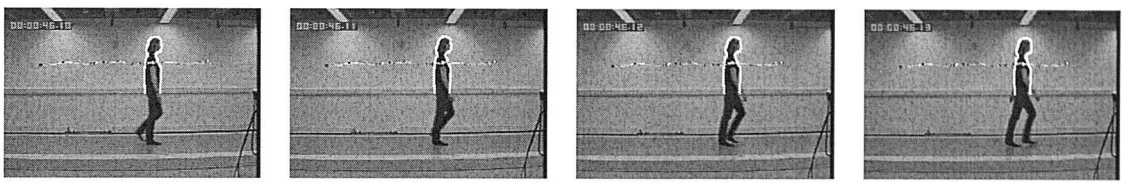


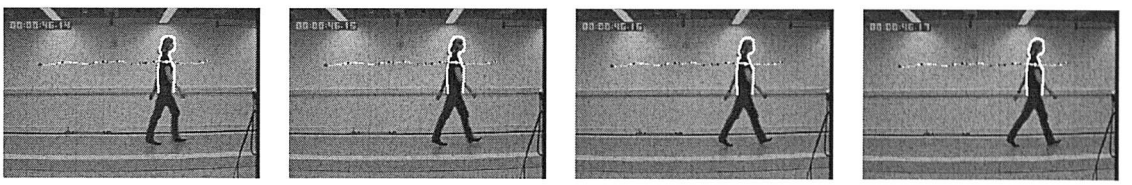
Figure 50: (part 1 of 2) MA3 extracted with the MA1 templates

Frame 24 Frame 25 Frame 26 Frame 27

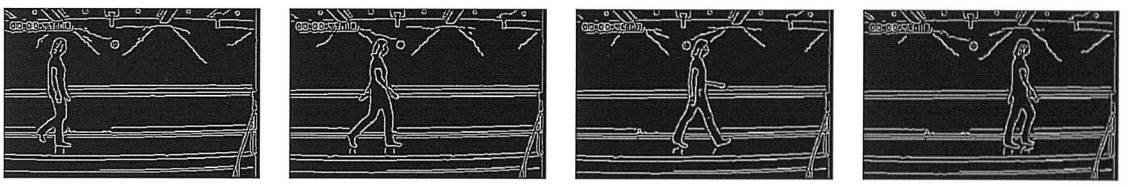
Frame 28 Frame 29 Frame 30 Frame 31



Frame 32 Frame 33 Frame 34 Frame 33



Frame 36 Frame 37 Frame 38 Frame 39

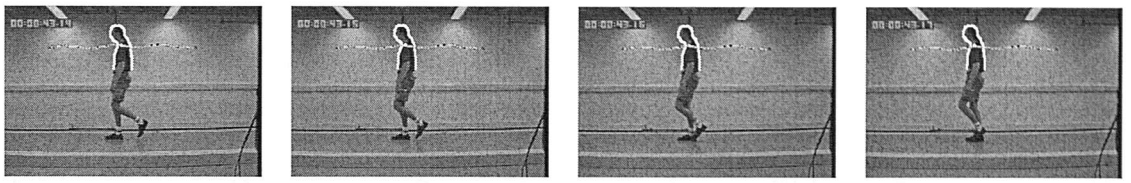


Edge-detected frame 05 Edge-detected frame 15 Edge-detected frame 25 Edge-detected frame 35

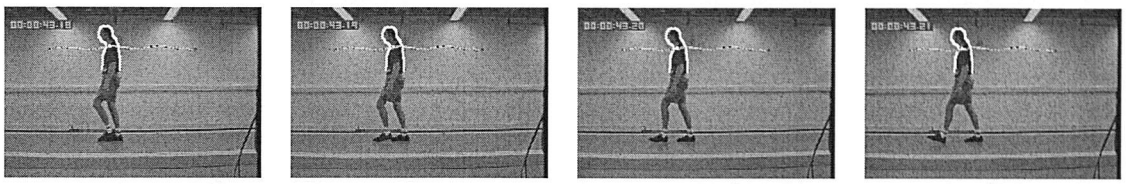
Figure 51: (part 2 of 2) MA3 extracted with the MA1 templates

9.3 SG1 extraction with SG1 templates

Figure 52: (part 1 of 2) SG1 extracted with the SG1 templates

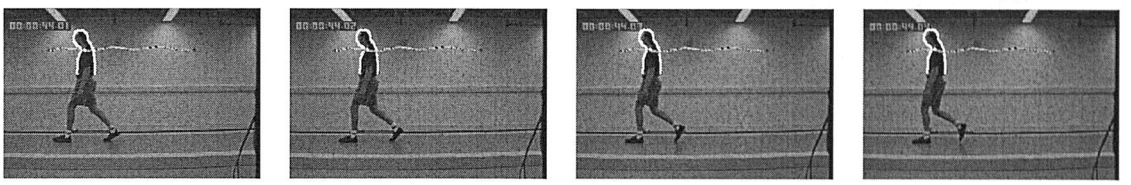


Frame 24 Frame 25 Frame 26 Frame 27

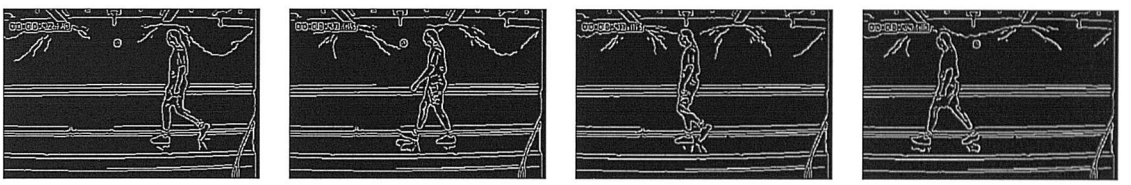


Frame 28 Frame 29 Frame 30 Frame 31

Frame 32 Frame 33 Frame 34 Frame 35



Frame 36 Frame 37 Frame 38 Frame 39



Edge-detected frame 05 Edge-detected frame 15 Edge-detected frame 25 Edge-detected frame 35

Figure 53: (part 2 of 2) SG1 extracted with the SG1 templates

9.4 SG3 extraction with SG1 templates

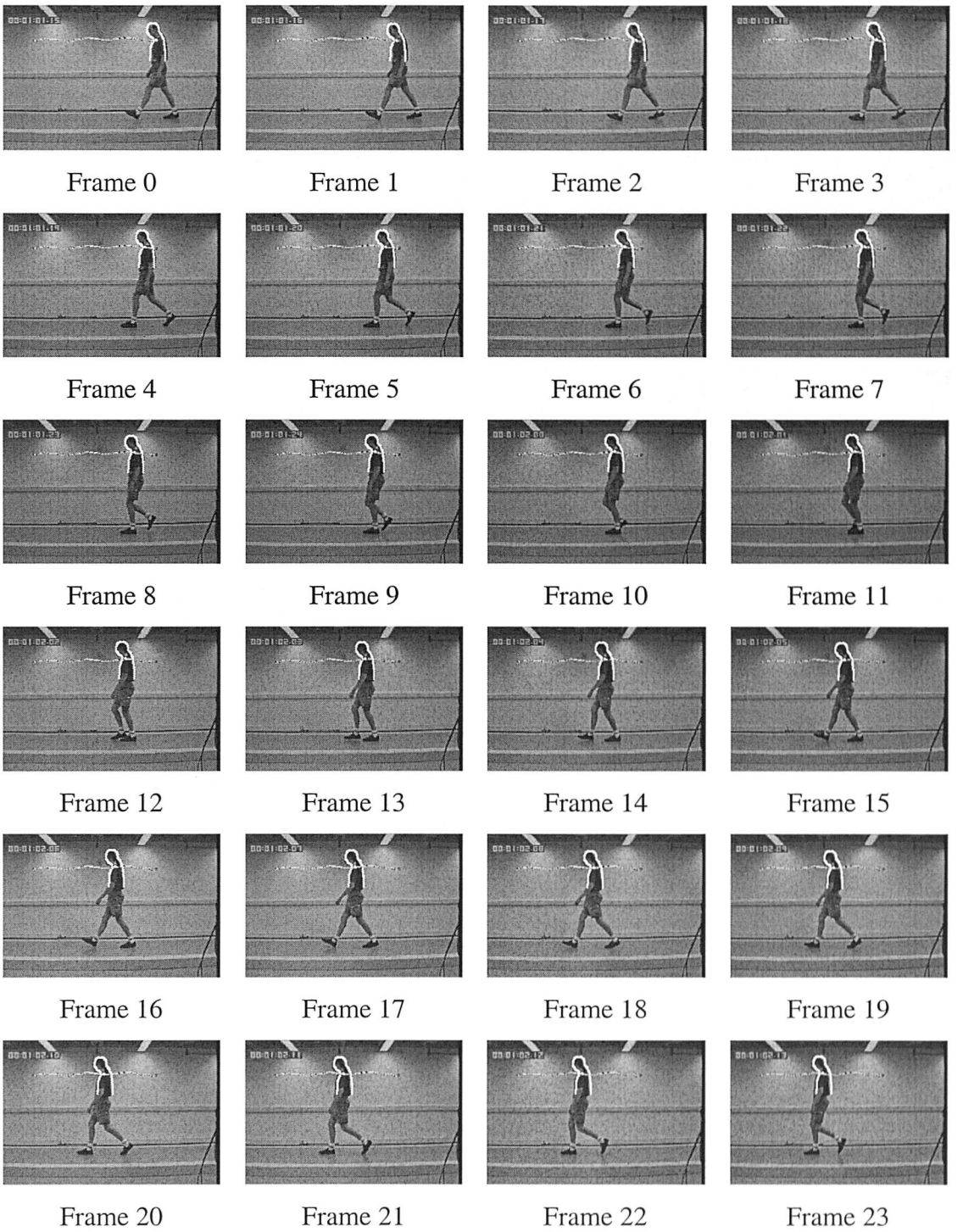


Figure 54: (part 1 of 2) SG3 extracted with the SG1 templates

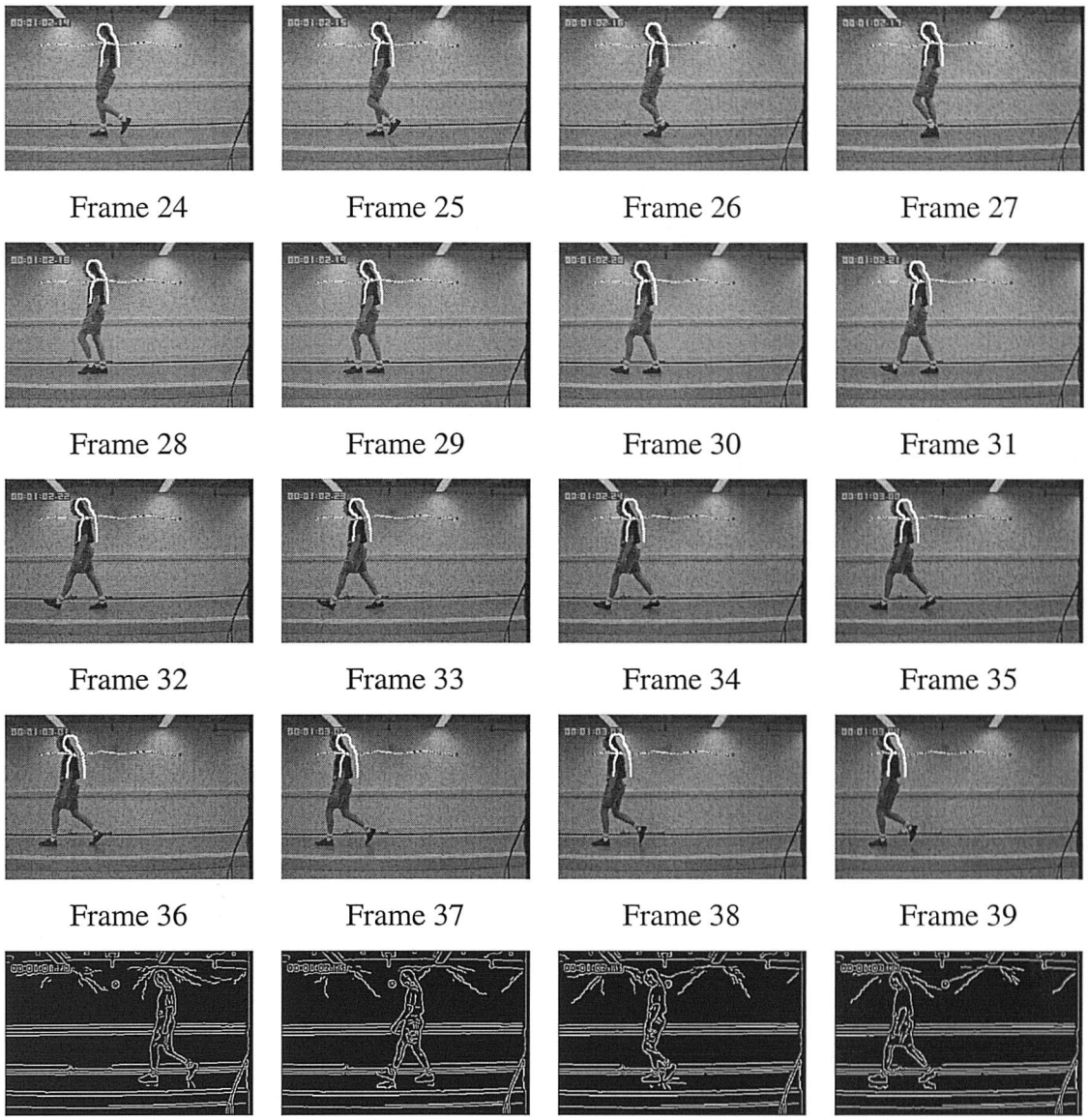


Figure 55: (part 2 of 2) SG3 extracted with the SG1 templates

9.5 VH1 extraction with VH1 templates

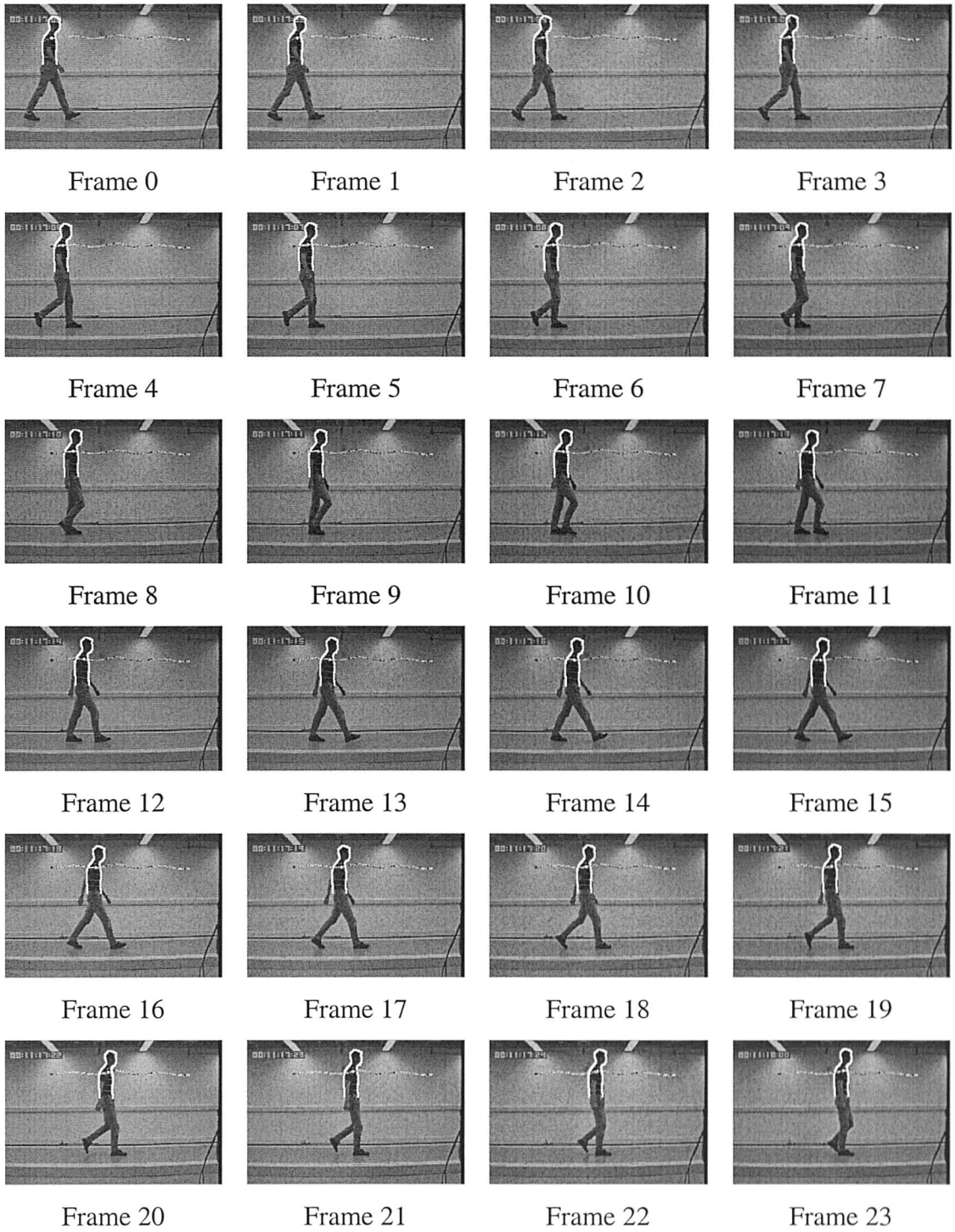


Figure 56: (part 1 of 2) VH1 extracted with the VH1 templates

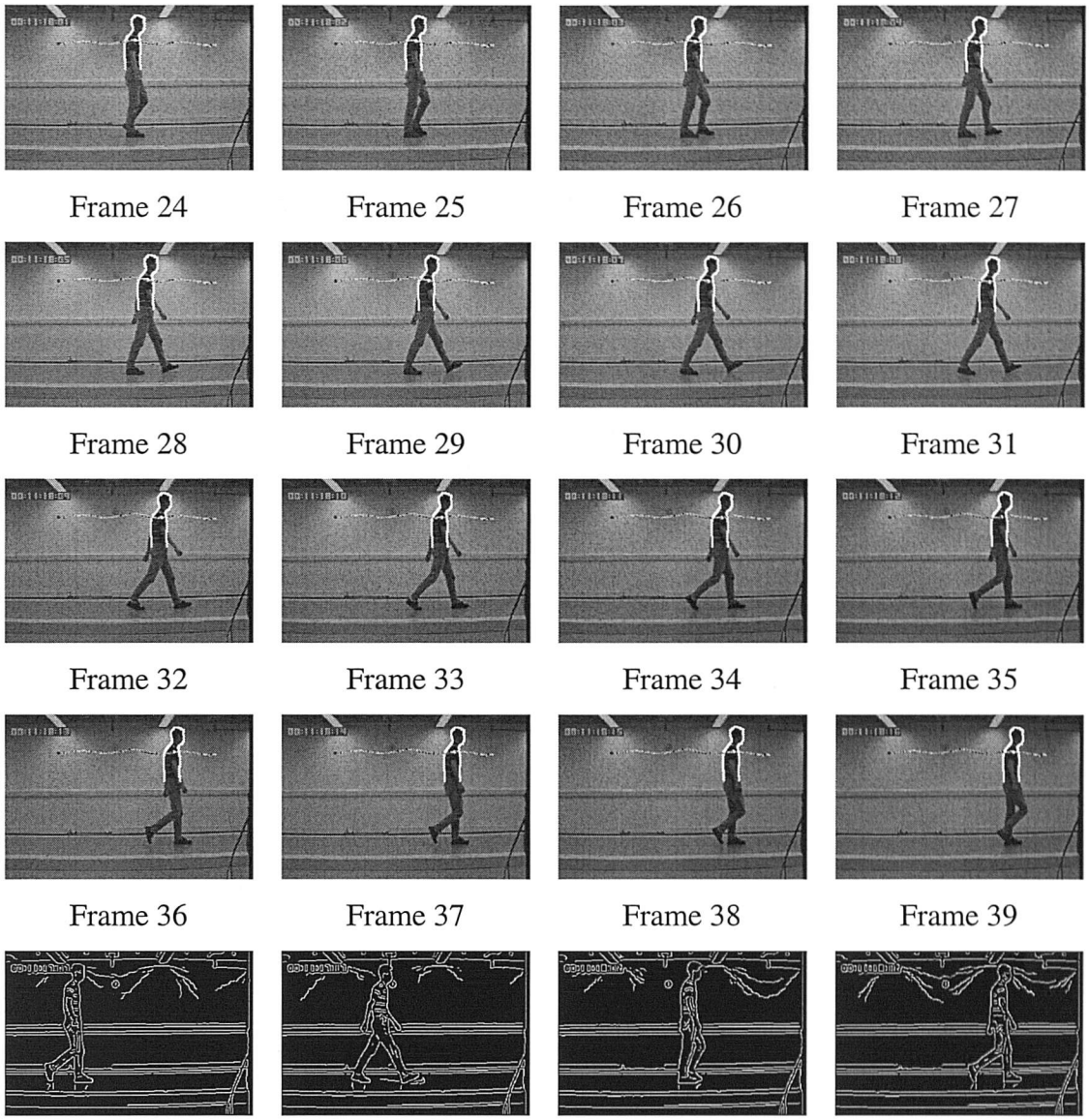


Figure 57: (part 2 of 2) VH1 extracted with the VH1 templates

9.6 VH3 extraction with VH1 templates

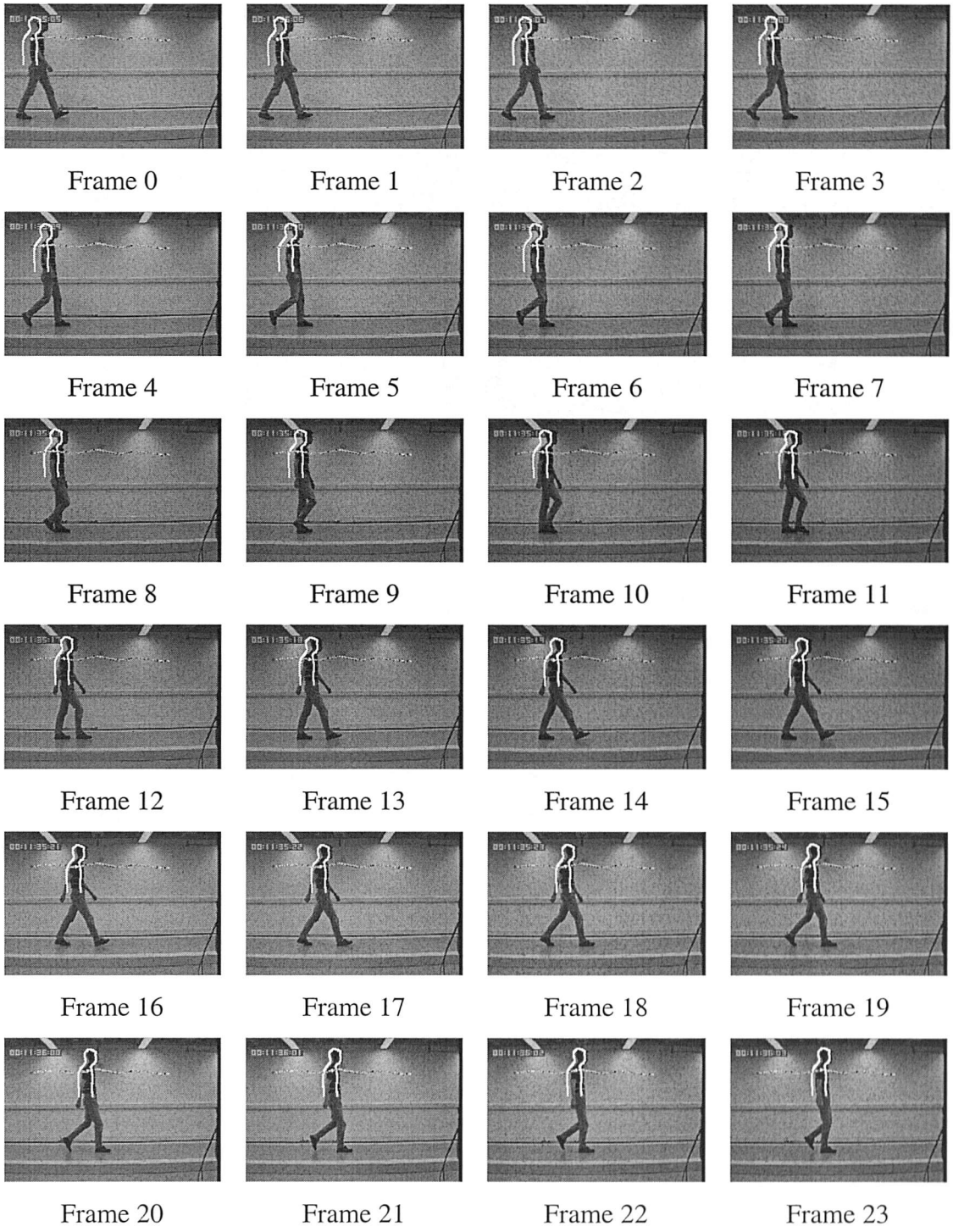


Figure 58: (part 1 of 2) VH3 extracted with the VH1 templates

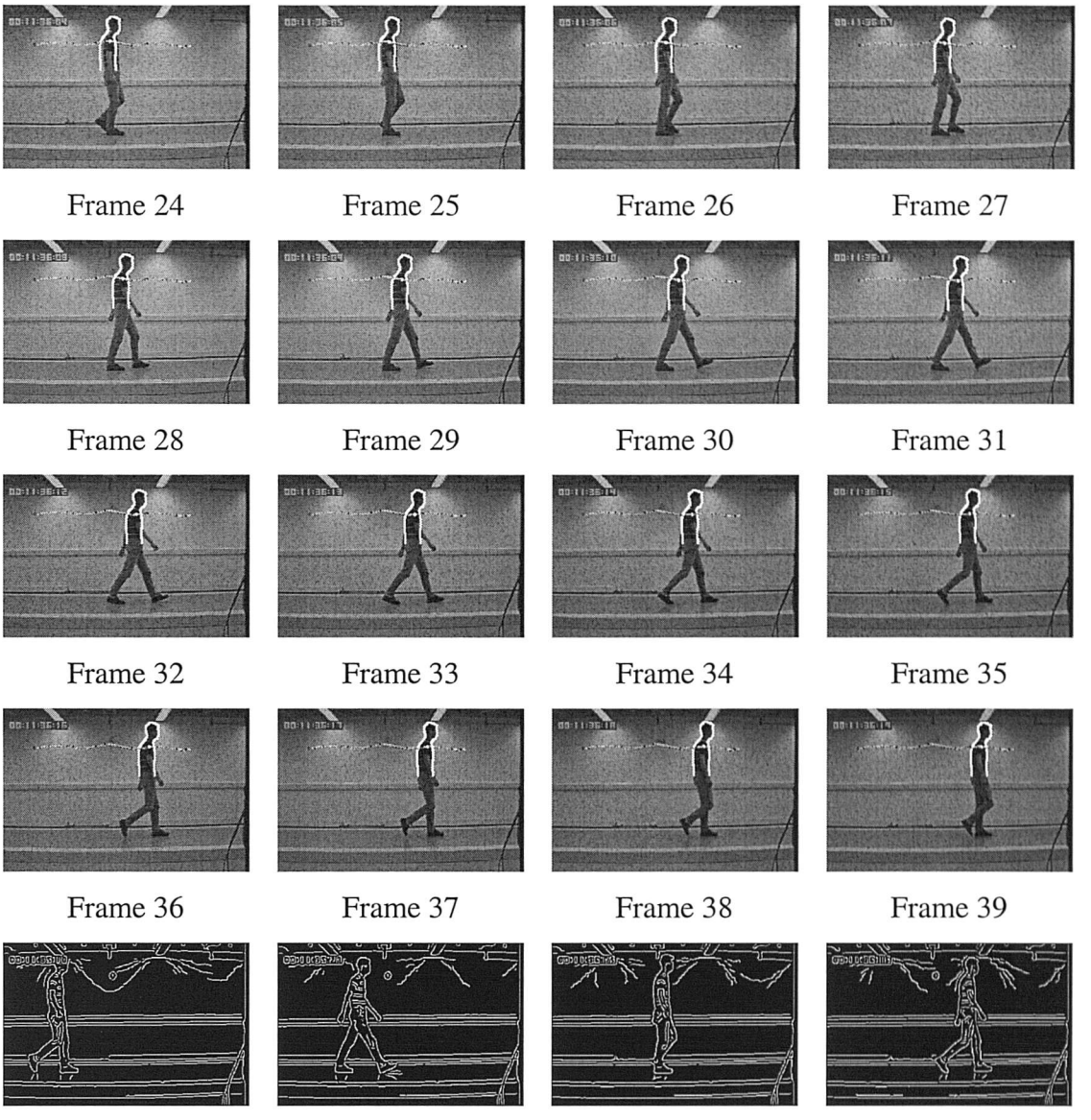


Figure 59: (part 2 of 2) VH3 extracted with the VH1 templates

10 Appendix: Pattern Recognition paper (final publisher copy)

Summary

There are currently available many approaches aimed at tracking objects moving in sequences of images. These approaches can suffer in occlusion and noise, and often require initialisation. These factors can be handled by techniques that extract objects from image sequences, especially when phrased in terms of evidence gathering. As yet, the newer approaches to arbitrary shape extraction avoid discretisation affects but do not include motion. The moving-object evidence gathering approach has yet to include arbitrary shapes and can require high order description for complex motions.

Since the template approach is proven for arbitrary shapes, we re-deploy it for moving arbitrary shapes, but in a way aimed to avoid discretisation problems. As the template approach has already been seen to reduce computational demand in the extraction of arbitrary shapes, we further deploy it to describe the motion of moving arbitrary shapes. As with the shape templates, we use Fourier descriptors for the motion templates, yielding an integrated framework for the representation of shape and motion. This prior specification of motion avoids the need to use an expensive parametric model to capture data that is already known. Furthermore, as the complexity of motion increases, a parametric model would require increasingly more parameters, leading to a rapid and catastrophic increase in computational requirements, whilst the cost and complexity of the motion template model is unchanged. The new approach combining moving arbitrary shape description with motion templates permits us to achieve the objective of low dimensionality extraction of arbitrarily moving arbitrary shapes with performance advantage as reflected by the results this new technique can achieve.

Extracting Moving Shapes by Evidence Gathering

Michael G. Grant, Mark S. Nixon* and Paul H. Lewis

Department of Electronics and Computer Science, University of Southampton, UK

[mgg | msn | phl]@ecs.soton.ac.uk | http://www.isis.ecs.soton.ac.uk/

Abstract

Many approaches can track objects moving in sequences of images but can suffer in occlusion and noise, and often require initialisation. These factors can be handled by techniques that extract objects from image sequences, especially when phrased in terms of evidence gathering. Since the template approach is proven for arbitrary shapes, we re-deploy it for moving arbitrary shapes, but in a way aimed to avoid discretisation problems. In this way, the discrete mapping operation is deferred as far as possible, by using continuous shape descriptions. A further advantage is reduction in computational demand, as seen in use of templates for shape extraction. This prior specification of motion avoids the need to use an expensive parametric model to capture data that is already known. Furthermore the complexity of the motion template model remains unchanged with increase in the complexity of motion, whereas a parametric model would require increasingly more parameters leading to an enormous increase in computational requirements. The new approach combining moving arbitrary shape description with motion templates permits us to achieve the objective of low dimensionality extraction of arbitrarily moving arbitrary shapes with performance advantage as reflected by the results this new technique can achieve.

Keywords: Arbitrary Shape Extraction; Motion Estimation; Tracking; Motion Template; Hough Transform; Evidence Gathering

***Address for Correspondence**: Department of Electronics and Computer Science, University of Southampton, Southampton. SO17 1BJ. UK.

1 Introduction

Tracking and motion estimation is an area of research that has a long history. There is a substantial literature on the problem of tracking shapes in a sequence, for example the tracking of humans has recently been admirably surveyed [**1**]. Earlier research has ranged from optic flow [**2**] to Kalman filters [**3**] and includes temporal templates, which we use here in an adaptation of the Hough Transform (HT) [**4**] to enable optimal extraction of moving arbitrary shapes.

Temporal templates are a technique for representing the movement of bodies through a sequence of images by encoding a motion trajectory. The encoding takes many forms - for example, there are many algorithms that combine spatial and temporal information into an XYT space to enable detection of particular movement patterns (e.g. detection of repetitive motion using temporal textures [**5**]). Other instances of the temporal template technique include a neural network based human motion tracker [**6**] that combines positional displacements with spatial templates of a human contour. The system contains several of these state vectors and is capable of tracking and predicting transitions between them. Similar efforts have been made using dynamic programming to track state transitions in gesture recognition [**7**].

In contrast, the thrust of this work is the *extraction* of moving shapes in an image sequence rather than tracking them. Extraction can be distinguished from tracking by a more holistic view of the sequence, and frequently use of *a priori* knowledge and an off-line approach. The new Velocity Hough Transform (VHT) [**8**] extracts conic sections moving in a parametrically described fashion. However, the VHT has two major restrictions: first, the shape description is limited to conic sections and, second, the motion description is parametric, causing computational difficulties as the complexity is increased.

Here, we propose a method that maintains optimal and efficient extraction of targets moving in a specified arbitrary fashion (via a temporal or "motion template") even in high levels of noise or occlusion. In addition, there is no need for initialisation or training. These qualities are derived from the use of robust evidence-gathering techniques and the exploitation of temporal correlation in the sequence.

In order to introduce motion templates into the HT, we have combined two other HT-related developments, integrating continuous template representations into a

temporal correlation framework. To shed light on the approach we have taken, there follow brief summaries of the main developments relevant to this work.

1.1 Generalised Hough Transform

Merlin and Farber [**9**] first considered general-shape detection using the HT but their method provided no means for detecting rotated or scaled shapes and was computationally expensive. Ballard developed the full mapping [**10**] for arbitrary shapes with rotation and scale invariance – the Generalised Hough Transform (GHT). The GHT replaces the analytic parametric constraints in the HT with a non-analytic tabular representation of an arbitrary shape. This table (the "R-table") describes the position of feature points in the template shape relative to a reference point and is indexed by the gradient direction information at each feature point. Compared with Merlin and Farber's method, this table increases the efficiency of the algorithm by reducing the number of feature points under consideration to those that fit an additional gradient direction constraint. Merlin and Farber trace entire instances of the template shape in the accumulator whereas the GHT only adds particular points from the template contour to the accumulator. In the GHT, the lower number of votes cast reduces the amount of noise in the accumulator generated by false votes (provided that the gradient data is of good quality) and can also improve the computational speed.

Early adaptations of the standard HT increased the complexity of the shape equation and thus the dimensionality of the HT – lines require a 2D parameter space, circles a 3D space, ellipses a 4D space, etc. Extrapolation suggests that for a parametrically defined arbitrary shape (effectively a high- or infinite-order polynomial) the standard approach would require a nearly infinite dimensional parameter space. In contrast with such an approach [**11**], which required an exorbitant accumulator space, an arbitrary shape HT actually only needs to accumulate for the (relatively few) appearance parameters, provided that the shape to be located is already specified. Instead of searching for the best fit to the parameters of an arbitrary polynomial, the only parameters that need examination are those that adjust the template shape to match the target – position, rotation and scale, for example. We will later see how templates can be used efficaciously not only in shape extraction, but also in motion extraction and description.

1.2 Fourier-descriptor template representation

In the GHT, the template shape is represented by an R-table, which is a discrete lookup table. The problems with this representation are well described in the literature (most recently and in greatest detail in **[12]**), but essentially derive from the fact that it is a discrete representation sampled at a particular scale. When the template is scaled or rotated, there can be problems with aliasing and rounding errors. **Figure 1b** shows the effects of scaling and rotating the discrete set of points comprising the original shape in **Figure 1a**. Clearly, the new sets of points have missing data (where the points in the original have become separated due to inadequate sampling at the new scale) and the effects of discretisation are evident. If the shape had been reduced in scale, the points in the original would merge, effectively oversampling the shape.

Distortions are inevitable when working with discrete systems. Nevertheless, the worst effects can be avoided by maintaining a continuous representation for as much of the process as possible. Using an analytically defined curve makes it possible to defer discretisation until after the rotation and scaling stages. Elliptic Fourier descriptors (FDs) **[13]** have been deployed in an adaptation of the GHT **[14]** to give such a continuous representation. Instead of recovering vote co-ordinates from an R-table, they are instead calculated from the FDs. This avoids the extra quantisation step inserted by the GHT (discretisation occurs in template transformations and again in the accumulation phase), thereby restoring the robustness of the original, analytic, HT formulation.

Clearly, if the original template is of a smaller scale than the reconstructed one, there will be no additional detail provided by the FDs. The FDs merely provide a continuous representation of the original template shape. Addressing the problem of disjoint or non-contiguous contours (i.e. one with gaps) can be achieved by using a masking function that causes the regeneration phase to ignore the points that would otherwise fill in the gaps.

Elliptic Fourier descriptors were chosen for their completeness, simple geometric interpretation, access to frequency information and the fact that they can be easily produced from a chain code of the contour. However, other analytic representations could equally have been used (e.g. cubic B-splines as in **[6]**).

1.3 Temporal evidence gathering

It is well known that most image sequences contain significant correlation between frames - a fact commonly utilised by machine vision and video compression algorithms amongst many others. The VHT [**8**] first exploited this correlation in evidence-gathering based techniques. The original implementation of the VHT extracted the optimum parameters describing a conic section moving with linear velocity. With simple extensions, it handles a subset of rigid motions that can be described parametrically - hence, the nature of the motion is known *a priori*. To take advantage of the inter-frame correlation, the VHT accumulates evidence from the whole sequence, concurrently extracting optimal structural and motion parameters. As a consequence of the additional information provided, the VHT is more robust than a standard frame-by-frame tracking implementation, especially when the target is occluded or noisy. Any missing or damaged structural information in a frame can be compensated for by redundant data in others (e.g. structural information in the target shape that is often repeated in each frame).

Due to the global scope of the VHT, there is no need to initialise the algorithm to search in a specific area (although limiting the extent of the search is a possible optimisation). Another common motion estimation problem avoided by the VHT is that of correspondence. Points in different frames do not need to be matched since all the possible correspondences are examined implicitly in the accumulation phase. By the nature of evidence gathering, the best correspondences produce the highest accumulator peaks.

The motion model is parametric and thus can be extended from linear velocity by including extra terms. In this respect, an extension to the VHT [**15**] to find walking subjects using an articulation model required thirteen parameters, when moving with constant velocity. The major disadvantage is that any extension to the motion model increases the dimensionality of the accumulator and, thus, the computational resources required. In summary, the VHT enables the use of temporal correlation in an evidence-gathering framework, resulting in a powerful and robust extraction algorithm. Unfortunately, the modelling of both shape and motion is seriously restricted.

1.4 Contributions

We will describe a new technique for extracting moving arbitrary shapes, which has been created by fusing the two evidence-gathering techniques, the VHT and Fourier-descriptor template representation (for convenience, we will refer to this continuous-template variant of the VHT as the CVHT). Uniting these techniques unifies their unique and complementary advantages. The Fourier descriptors provide a continuous template representation, minimising discretisation error in the algorithm, and the VHT component exploits the temporal correlation across a sequence, mitigating the effects of noise and occlusion. The new algorithm does not require initialisation or training and avoids the need to solve the correspondence problem, inheriting these characteristics from the VHT. For illustration, **Figure 2** shows frames of a sequence where the location, velocity and acceleration parameters of a Space-Shuttle booster during launch were correctly extracted by the new technique.

However, the CVHT is still limited by its parameterised motion model. If the shuttle imagery included its parabolic trajectory, this more complex motion would have to be incorporated into the accumulation phase, requiring many more parameters. Hence, the simplistic approach to improving the generality of the motion model is to increase the complexity of the HT kernel to represent an increasingly complex motion path. Consequently, an accurate polynomial description of an arbitrary path will require a large or even infinite number of terms, massively increasing the dimensionality of the problem. There are parallels to this parameterisation of motion in the earlier parameterisation of shape, where increasingly complex shapes were represented by more complex parameterisations and a commensurately larger dimensionality. The solution to this dimensional explosion was found in the use of templates, which allowed an efficient and low dimensional parameterisation of any shape. The cost of this approach is that the method loses the (debatable) flexibility of finding all descriptions of all possible shapes in a scene. Following this historical parallel, the other part of our new approach is to describe the motion by a template, like the shape itself. These "motion templates" extend the use of templates in the HT from the spatial domain into the temporal. This ameliorates the punitive computational burden associated with increasing dimensionality since the aim changes from finding the potentially

unlimited set of parameters that characterise a particular motion to finding the limited set of parameters that locate the object undergoing the specified motion.

These novel developments clearly address two core weaknesses of the VHT – namely that it is limited to extracting analytically described conic sections only and that the motion model is parametric. Adding the capability to extract non-analytic arbitrary shapes that move arbitrarily increases the utility of the algorithm to a range of applications that require more general shape and motion models.

We first describe our approach to arbitrary shape extraction with parametric motion (Section 2.1) and with arbitrary motion (Section 2.2). We next present results of some comparative evaluation and performance analysis in Section 3. Finally, we give conclusions and suggest directions for future research.

2 Extracting Arbitrarily Moving Arbitrary Shapes

2.1 Continuous Velocity-HT: Extracting Moving Arbitrary Shapes

Effectively, the Fourier descriptor version of the GHT is extended in the same way as the HT for circles was extended into the VHT - by introducing velocity terms to the shape description. Instead of drawing a motion-compensated circle in the accumulator (as in the VHT), the Fourier descriptors are used to trace a locus of votes in the form of the template shape, adjusted for the estimated motion of the object relative to the time reference of each frame. The accumulation process (Merlin and Farber's variant) for the sequence in **Figure 3a** is illustrated in **Figure 3b-d** with votes increasing as more frames are added. Flipped instances of the template shape are generated in accumulator, centred on motion-compensated edge pixel co-ordinates from the frame being processed. The motion compensation is simply back-projection along the expected line of motion to convert the co-ordinates to the same temporal frame of reference as the initial frame, meaning that frame 5's votes for a particular edge pixel fall in the same place as those for frame 1. Once the voting process is complete, some intersections of template shapes in the accumulator form peaks that indicate the location (at time $t = 0$) of an instance of the shape in the sequence.

In order to develop the voting mechanism, we require an arbitrary-curve parameterisation for shapes. A curve defined by two sets of orthogonal co-ordinates, $c_x(s)$ and $c_y(s)$, parameterised by $s \in [0,2\pi)$ has elliptic Fourier descriptors as follows:

$$a_{xk} = \frac{1}{\pi}\int_{-\pi}^{\pi} c_x(s)\cos(ks)\mathrm{d}s \quad \text{and} \quad b_{xk} = \frac{1}{\pi}\int_{-\pi}^{\pi} c_x(s)\sin(ks)\mathrm{d}s \qquad (1)$$

with a similar equation for the y descriptors, where k is the harmonic number. The range of k defines the number of ellipses used to represent a model shape and thus how accurate the shape representation is. In the general case, using more harmonics gives a more accurate representation of a shape. However, since the FDs are used in the context of a digital (thus discrete) image, there will be a limit defined by the pixel granularity beyond which additional harmonics have no perceivable effect. Before FDs can be used to draw the shape in the accumulator, they must be converted from the frequency domain to vectors in the spatial domain that specify co-ordinates on the contour of the template shape. n FDs can be converted to vectors (along x- and y-axes) from the origin to a point on the curve by:

$$\upsilon_x(s, \overline{FD}_x) = \sum_{k=1}^{n} \left(a_{xk} \cos(ks) + b_{xk} \sin(ks) \right) \quad (2)$$

where $\overline{FD}_x = \{a_{x1}, b_{x1}, a_{x2}, b_{x2}, ..., a_{xn}, b_{xn}\}$, with complementary equations in y. The DC terms have been omitted, which translates the curve so that its centre is at the origin. The shape's initial scale and rotation is given by $\mathbf{a}_s=[l_g \ \rho_g]$ and the scaled and rotated shape itself can be described as

$$R_x(s, \mathbf{a}_s) = l_g \upsilon_x(s, \overline{FD}_x) \cos(\rho_g) - l_g \upsilon_y(s, \overline{FD}_y) \sin(\rho_g) \quad (3)$$

with a similar equation for R_y. Now, we require a kernel that defines the shape of votes to be laid down in the accumulator for each feature point (e.g. an edge pixel). This is a combination of curves, each with its origin on the reference point to be voted for (typically at the centre of the template shape) but offset by the velocity, and at a number of orientations and scales (for similarity transform invariance). This combination of curves can be obtained from:

$$\overline{\omega}(s, f, l, \rho, v_x, v_y) = R_x(s, l, \rho) U_x + R_y(s, l, \rho) U_y + f v_x U_x + f v_y U_y \quad (4)$$

where v_x and v_y are respectively the x centre and y centre velocity parameters and U_x and U_y are two orthogonal vectors defining the x- and y- axis respectively. This curve is inserted into the accumulator by offsetting it from the co-ordinates of each feature point in the image sequence IS, defined by:

$$IS = \{\overline{\lambda}(t, f) \mid t \in D_t, f \in D_f\} \quad (5)$$

Here, $\overline{\lambda}(t, f)$ is a parametric function that defines the points in the image sequence for a frame time f, where a suffix on the domain indicates its extent (here, D_t is the domain of an image in the sequence and D_f is the domain of the frames of the sequence). The accumulator vote-pattern expression is then:

$$A_t = \{\overline{\lambda}_{t,f} - \overline{\omega}(s, f, l, \rho, v_x, v_y) \mid s \in D_s\} t \in D_t, f \in D_f \quad (6)$$

These equations describe the concept of the HT but do not formalise the actual technique used, namely the accumulation phase. The parameter space can be mapped into an accumulator by using a matching function, which determines whether a point, $\bar{c}$, in parameter space should be incremented for a point, $\bar{d}$, in the set A_t. The equation below defines the simplest accumulation strategy, namely incrementing an accumulator cell by unity for each match. Changing the matching function M can accommodate more complex strategies.

$$M\left(\bar{c},\bar{d}\right)=\begin{cases}1 & \text{if } \bar{c}=\bar{d}\\ 0 & \text{if } \bar{c}\neq\bar{d}\end{cases} \tag{7}$$

Next, this function is applied to A_t for a range of parameter values. This defines the continuous form of the CVHT, accumulating evidence in a parameter space S_{CVHT} according to:

$$S_{CVHT}\left(\bar{b},l,\rho,v_x,v_y\right)=\iiint M\left(\bar{b},\ \bar{\lambda}(t,f)-\bar{\omega}\left(s,f,l,\rho,v_x,v_y\right)\right)ds\ dt\ df \tag{8}$$

Where $\bar{b}$ is the translation vector (i.e. the location of the reference point). Finally, this parameter space is sampled into a discrete multidimensional array S_{DCVHT}, which is expressed by:

$$S_{DCVHT}\left(\bar{b},l,\rho,v_x,v_y\right)=\sum_{f\in D_f}\sum_{t\in D_t}\sum_{s\in D_s}M\left(\bar{b},\ \bar{\lambda}(t,f)-\bar{\omega}\left(s,f,l,\rho,v_x,v_y\right)\right) \tag{9}$$

This expression gives an accumulation strategy for finding moving arbitrary shapes. For each edge pixel in a frame, a locus of votes is calculated from the Fourier description of the template shape and entered into the unified accumulator. The co-ordinates of the loci are adjusted to allow for the predicted motion of the shape, dependent on the frame index, as in the VHT.

2.2 Motion Templates: Extracting Arbitrarily Moving Shapes

2.2.1 Overview

The earlier approaches to moving shape extraction are limited by the dimensionality of the parameterised motion model. As stated, our objective is to remove this limitation and enable the extraction of arbitrary shapes undergoing arbitrary motion by using motion templates to describe the movement of the target. With these templates it is no longer necessary to accumulate for the parameters describing the motion since they are already known. It may be useful to imagine the motion template in terms of an infinite dimensional parametric motion model where all the parameters have a fixed value. Naturally, the motion templates require *a priori* knowledge concerning the target object's path before analysis can begin. Since, by definition, tracking precludes the possession of this information, it is important to observe that the niche for motion templates in the context of evidence gathering is extraction or recognition (an example application might be searching a video database). Motion templates make it possible to robustly and efficiently extract parameters describing a shape that is following a specified trajectory. A limited prediction capability is possible if the motion template covers a larger timespan than is analysed (or in the case of repetitive motion) since it is assumed that the extracted subject will continue to follow the specified path.

As a result of the requirement for detailed prior knowledge, the new algorithm will be of use in cases where the general path of motion is known (e.g. cars turning at traffic lights will follow roughly the same path). However, there may be difficulties in real world imagery since not all objects will follow exactly the same path. The forgiving nature of an evidence-gathering approach should abate this concern provided the deviations are not excessive. If they are, the voting or peak detection algorithms can be arranged to handle the uncertainty, (e.g. as in the Fuzzy HT [**16**]).

Motion templates must encode the relative position of the target object at all times. This will automatically describe properties of the motion such as speed, acceleration, change in acceleration, etc. Motion templates should also record changes in scale and rotation over time since, in motion, many objects rotate (e.g. a car when viewed from above rotates as it corners) and scale (e.g. due to perspective effects). Note that this additional detail will not cause any increase in accumulator dimensionality – the

complexity of the algorithm is the same, only the complexity of the motion template itself has increased. It is possible to record more information relating to time-structured changes in the model shape (e.g. changing models as the sequence progresses to represent deformations) but position, scale and orientation are a natural minimum.

The representation of the motion templates should be continuous so as to avoid the problems of discrete representations (section 1.2). As such, it seems prudent to use Fourier descriptors for both motion and shape templates, since these descriptors are well understood and Fourier approaches can handle many situations (e.g. non-regular path sampling). A further advantage is access to frequencies in the motion template, which may be of use in certain applications. Furthermore, we have a consistent framework for the representation of arbitrary shape and motion.

To encode the path for input to an implementation, it is convenient to specify a series of waypoints to encode the path, rather than use a smooth and complete description of the motion. The representation chosen must be able to take this data and interpolate it in a smooth fashion. Fourier descriptors have been designed to do this from the start and only need minimal modification to work with variable time periods between waypoints (required since movements may be quick or slow). The main danger comes from under-sampling the path with too few waypoints. In this case, it is possible to over-fit the FD and reproduce this under-sampled path too exactly. If desired, the waypoints could be filtered to generate a more smoothly flowing path.

The introduction of motion templates into the HT requires no additional parameters to be searched for in the accumulation phase. However, such an inflexible implementation would restrict the functionality excessively. Essential parameters are rotation of the motion template in its spatial dimensions and scaling in both spatial and temporal dimensions. The scaling in the spatial axes does not need to be independent (i.e. it can be uniform scaling) since we will only be dealing with affine transformations initially. Scaling in the temporal axis adjusts the time taken to traverse the motion template and thus the speed with which an object must move to be identified as the target. Using the previous example, this would allow the algorithm to locate cars cornering quickly or slowly. Finally, we must add an offset or phase parameter to separate the time encoded in a motion template from that used in a sequence. Otherwise, the frame time zero would be tied directly to time zero in the

motion template and the algorithm would be unable to correctly extract parameters for a sequence that began with the target object part-way through its trajectory. Hence, the time-scale parameter provides temporal scale invariance and the offset parameter gives temporal translation invariance. Rotation invariance is not required since time is one-dimensional.

Accordingly, the use of motion templates adds four extra parameters, giving an eight–dimensional accumulator. While this will cause a large increase in the computational requirements, it is much less than the dimensional explosion presented by polynomial extension. Clearly, the ability to extract, optimally and robustly, arbitrary shapes following an arbitrary path is well worth the additional computational resources.

2.2.2 Theory

The theory supporting motion templates is developed from the CVHT, which itself is an extension to the Fourier-descriptor variant of the GHT [**14**]. As an aid to visualisation, for a given feature point (e.g. an edge pixel in a frame), a locus of points is plotted through the eight-dimensional accumulator space. This locus is constructed from the shape and motion descriptors, which are used to trace scaled and rotated instances of the template shape, translated to compensate for the expected motion of the object relative to the time reference of each frame, in two-dimensional planes taken along the x- and y-axes of the accumulator. Once the voting process is complete, peaks in the accumulator indicate the location and other parameters (at time $t = 0$) of an instance of the target shape moving along the specified path.

The changes to the CVHT are confined to implementing motion templates as the motion model. Hence the shape parameterisation is unchanged and Equations 1 through 3 in section 2.1 describe the shape template. Having parameterised the shape template, we must now parameterise the motion template, encoding the changing x, y, rotation and scale parameters. There are two major differences in the definition of the motion template from that of the shape parameterisation. The first is that the DC terms of the FDs are retained and used in the reconstruction. Removing them would effectively translate reconstructed co-ordinates relative to some arbitrary origin (e.g. the centre of the template), which would destroy the utility of the sequence by moving the start point. Secondly, and relevant in terms of implementation or specifics of template definition only, the offsets used are not derived from chain codes, which

have fixed x, y and t values, but come from a list of waypoints and thus can instead specify larger increments.

Let $m_x(f)$, $m_y(f)$, $m_p(f)$ and $m_l(f)$ be functions that take a frame time f and recover a co-ordinate (for x position, y position, rotation or scale respectively) from a motion template. This motion template is now used to transform the co-ordinates calculated from the shape descriptors. These co-ordinates have already been globally scaled and rotated (i.e. we are using R_x and R_y from Equation 3) to adjust for possible initial scales and orientations. They are then scaled and rotated as:

$$\xi_x(s,f,o,\mathbf{a}_s) = m_l(f-o)R_x(s,\mathbf{a}_s)\cos(m_\rho(f-o)) - m_l(f-o)R_y(s,\mathbf{a}_s)\sin(m_\rho(f-o)) \quad (10)$$

with a similar equation for y co-ordinates and where f is the frame of the current image and o is a parameter that offsets any mismatch between the frame times in the sequence and in the motion template. $m_l(f\text{-}o)$ and $m_p(f\text{-}o)$ recover scale and rotation, respectively, for time f - o from the motion template. These rotated and scaled co-ordinates now represent the shape at the expected orientation and scale. Next, they are translated to compensate for the object's expected motion. However, the path of expected motion is also scaled and rotated (requiring parameters, $\mathbf{a}_m = [l_m \quad \rho_m]$):

$$\mu_x(s,f,o,\mathbf{a}_m,\mathbf{a}_s) = \xi_x(s,f,o,\mathbf{a}_s) + l_m m_x(f-o)\cos(\rho_m) - l_m m_y(f-o)\sin(\rho_m) \quad (11)$$

again with a similar equation in y and where $m_x(f\text{-}o)$ and $m_y(f\text{-}o)$ are the x and y offsets recovered from the motion template at time f - o. Now we can form the kernel that defines the shape of votes in the accumulator. This is a multi-dimensional combination of the template at a number of translations, orientations and scales, and can be obtained from:

$$\overline{\omega}(s,f,o,l_f,\mathbf{a}_m,\mathbf{a}_s) = \mu_x(s,(f\cdot l_f),o,\mathbf{a}_m,\mathbf{a}_s)U_x + \mu_y(s,(f\cdot l_f),o,\mathbf{a}_m,\mathbf{a}_s)U_y \quad (12)$$

One final parameter, l_f, has been added to perform temporal scaling on the motion template, thereby allowing adjustment of the speed at which the path is traversed. To ensure that the locus drawn in the accumulator passes through the reference point, the kernel is offset from the image co-ordinates of each feature point. Hence, for an image sequence IS (defined in Equation 5), the votes are placed in the accumulator as:

$$A_t = \{\overline{\lambda}(t,f) - \overline{\omega}(s,f,o,l_f,\mathbf{a}_m,\mathbf{a}_s) \,|\, s \in D_s\} \forall t \in D_t, f \in D_f \quad (13)$$

As with the CVHT, the parameter space formed by the application of the expression above is mapped into an accumulator by the use of a matching function

(Equation 7). This multi-dimensional accumulator space is sampled into a discrete parameter space S_{DMT} given by:

$$S_{DMT}(\bar{b}, l_f, o, \mathbf{a}_m, \mathbf{a}_s) = \sum_{s \in D_s} \sum_{t \in D_t} \sum_{f \in D_f} M(\bar{b}, \bar{\lambda}(t, f) - \bar{\omega}(s, f, o, l_f, \mathbf{a}_m, \mathbf{a}_s)) \quad (14)$$

where $\bar{b}$ is a vector of the image co-ordinates at time 0, l_f is the time scale-factor, o is the time offset (phase) and $\mathbf{a}_m$ and $\mathbf{a}_s$ contain scale and rotation parameters that respectively transform the motion template's path and transform the initial orientation of the target shape. $\bar{\lambda}(t, f)$ retrieves feature points from the sequence. This expression gives an accumulation strategy for finding arbitrary shapes moving arbitrarily. It allows extraction of the optimal parameters describing an arbitrary (but specified) shape of unknown orientation, position and scale that is following an arbitrary (but specified) path of unknown orientation and scale, which takes an unknown time to traverse.

The new algorithm currently traces the entire template shape in the accumulator for each feature point and for each parameter combination. The GHT places a restriction on which template points are drawn - only those with the same gradient direction as the edge pixel being processed are added to the accumulator so that only the relevant fractions of the template are traced. With accurate gradient direction data, this restriction removes a lot of unnecessary votes (and hence noise) from the accumulator. The voting algorithm could be changed to perform the same reduction of votes as the GHT by incorporating a function that calculates the gradient direction at a point on the Fourier-described curve. This value would then be compared against edge pixel gradient direction to restrict the votes cast into the accumulator. Depending on the computational cost of the gradient direction calculation, this reduction in voting may result in a linear speed improvement.

2.2.3 Computational cost

The algorithm is of order *O(#points * #s * #r * #mt_s * #mt_r * #t_off * #t_s)*, where *#points* is the number of feature points (e.g. thresholded edge-pixels) in the sequence and *#s*, *#r*, *#mt_s*, *#mt_r*, *#t_off* and *#t_s* are the number of discrete steps in the parameter ranges for initial shape scale and rotation, scale and rotation of the motion template and temporal offset (phase) and scaling respectively. Hence, the new development inherits the usual computational cost penalty of the HT: the accumulator is eight-dimensional and can require significant resources. This is unavoidable if the

algorithm is to be used in its most basic form since, in the HT, the process that guarantees optimality is an efficient form of exhaustive search. However, most of the speed-up and memory-reduction modifications to HT-related techniques [**4**] are applicable – e.g. parameter space decomposition, multi-stage processing, random sampling algorithms or genetic algorithms. Also, the HT is known to be well suited to parallel implementation, with multiple opportunities for splitting the processing (e.g. by image, by region of an image, by ranges of parameter values, etc).

2.2.4 Example

As an example of the information that may be encoded in a motion template, **Figure 4** shows the changes in the x and y positions and the angle of the left foot of the walker pictured as the sequence progresses. The measurements were made manually with the x and y positions being taken from the heel of the foot and the angle being estimated by eye. This illustrates the information that a motion template might have to record, in this case, for extracting the location of a foot (a better description of the gait cycle may be found in [**17**]). Unfortunately, it is difficult to produce a more direct visualisation of a motion template because it must encode multiple dimensions of data (e.g. x, y, rotation, time), not to mention the possibility of the path looping back on itself.

3 Results

3.1 Continuous Velocity HT

3.1.1 Gaussian noise testing

The CVHT was run on a five-frame sequence based on a small (20×20) image (the first image in **Figure 5**) moving linearly along the x-axis at a velocity of one pixel per frame. A small image was chosen to make practical computation of large-scale tests. Noise was added at random to each frame of the sequence at eleven noise levels from 0% random coverage to 100% random coverage of the frame. The noise distribution was zero-mean Gaussian with a standard deviation of three and clipped when a pixel value exceeded the maximum allowable. Examples of the effects of the increasing noise levels can be seen in **Figure 5**. The grey-level images produced are thresholded by the algorithm and are shown in the thresholded state. The results for the normal GHT tracking algorithm were generated using a standard GHT on each frame of the sequence and using linear regression on the results to calculate the velocity terms. The test conditions were as described above.

The graph (**Figure 6**) shows the CVHT is significantly more accurate than the GHT-based technique. By accumulating temporally correlated evidence the new technique is able to handle noise levels that are approximately twenty percent greater than the standard. The GHT based technique is limited to the amount of evidence available in a single frame. When the noise becomes sufficient to mask out the correct peak in a single frame, the GHT technique is left with effectively random results to process for velocity terms. In intermediate noise levels, some of the results for each frame may be incorrect and this will tend to throw off the final regression step. The integrated approach taken by the new algorithm is more global and is not so susceptible to corrupt frames as is demonstrated strongly in the occlusion testing later. This is consistent with earlier observations comparing the VHT to the HT with linear regression [8].

3.1.2 Occlusion testing

A simple test of the effects of occlusion was carried out on the five-frame sequence described above. No noise was added to the sequence since this would add an unnecessary factor. Instead, a number of vertical lines of pixels starting from column

two were blanked out and both algorithms run on the resulting image. **Figure 7** shows example frames from the occluded sequence where the occlusion bar is five pixels wide (the bar is shown in grey to make it visible).

The results shown in **Figure 8** reveal that the new technique keeps track of the shape until the blanking is fifteen pixels wide – which obscures the shape for the duration of the entire sequence. The GHT based algorithm failed as soon as any blanking was introduced. This failure reveals more about the algorithm's implementation than about its resilience to occlusion. The current implementation uses the estimated location of the template shape in every frame as an input to the linear regression stage. Therefore, when a frame is corrupted and gives an incorrect result, the output of the linear regression stage is affected causing a global estimation error. A more sophisticated implementation might include a heuristic that ignores frames giving evidence inconsistent with the majority of frames.

The earlier results [**8**] relating to the VHT should also be applicable to the new technique since the underlying characteristics are essentially unchanged. These results indicate that VHT derived algorithms are capable of handling even extreme occlusion due to the global integration of evidence across the entire sequence.

3.1.3 Finding People with the CVHT

For purposes of illustration, the CVHT is now applied to locate a moving human body in a sequence of images. The current implementation of the new technique locates rigid shapes moving with linear velocity. Clearly, its formulation is general so shape deformation could be included, as it was for pulsating arteries in the original VHT formulation [**8**], but this would be considerably more complex. In the case of a human walking, the torso is approximately a constant shape and, if the camera is far enough away, the bobbing motion of gait is small enough to be compensated for by the resilience of the evidence gathering approach. Consequently, it is possible to detect people using the technique in its current form by searching for the torso. However, no meaningful gait data can be gathered from just the location of the torso so this method of locating a human silhouette is only useful as a primer for later statistical (gait) analysis. Nonetheless, using the CVHT to locate a human demonstrates that it is equally applicable to real world images.

Self-occlusion of the body due to the motion of the arms and legs is a problem that affects the performance of many person-tracking algorithms. By the nature of

evidence gathering, the new algorithm copes with occlusions that do not reduce the number of correct votes (from the uncorrupted remainder of the true contour across the whole sequence) below the level of noise in the image. As a result, there is no immediate need for special precautions.

Figure 9 shows a reconstructed template of a walker's torso, which was originally created by manually tracing the torso in the first frame of the sequence. Also shown in **Figure 9** are several frames of the MP1 walker sequence with the template superimposed on the extracted locations. During the first part of the sequence, the walker's location is accurately extracted - the initial location is exact and the extracted speed (thirteen pixels per second) is correct. Shortly after frame seven the walker rises up on his leg (vertically, there is a rise of fifteen pixels), which will cause the votes to "miss" in the accumulator, since this movement has not been accommodated in the evidence gathering strategy. He also slows down slightly and this breaks the assumption of linear velocity and consequently the template "overtakes" the walker - frame 10 shows it some pixels ahead.

If noise is added, as in section 3.1.1 or 3.1.2, the performance drops off rapidly due to the large number of missed votes arising from the lack of modelling of the bobbing or vertical motion of gait. Under simulated addition of noise, as in **Figure 6**, the decline in accuracy is similar to that of the synthetic sequences but drops away at a lower noise level since the level of background noise is higher due to the surrounding scenery.

3.2 Motion Templates

It is usually prudent to compare new techniques with contemporary or equivalent approaches. This appears unfeasible in this case since the most appropriate comparator technique is the GHT, but with interpolation, or tracking. The suitable form of this interpolation for the GHT or the motion model for tracking is actually the motion template, the very subject of these results. The alternative would be a fully representative parametric motion model that, as explained previously, is computationally intractable due to its infinite dimensionality. Consequently, the comparison that would be made is that of a frame-by-frame extraction process with a non-analytic template representation against an integrated multi-frame extraction process with an analytic template representation. This comparison has already been made in section 3.1, which examined the earlier approach to moving arbitrary-shape extraction (without motion templates but with a linear motion model) and consequently fails to test the subject of this section - the motion template in an evidence gathering context.

Comparison with other techniques that use similar knowledge of motion (such as the neural network based human motion tracker **[6]** or the spatio-temporal repetitive motion detector using temporal textures **[5]** referred to in the introduction) is not comparing like with like. In the case of techniques that are as dissimilar as a tracker and an extractor, the comparison is best made on application-dependent qualitative requirements or on the basis of each technique's features (e.g. optimality vs. on-line performance), rather than a quantitative performance analysis. In light of these difficulties, we have examined the performance of the new technique in terms of noise affecting each component of the system rather than attempt to make direct comparisons with other distantly related approaches. We believe that such analysis will enable the aforementioned choice based on requirements or features.

3.2.1 Image-noise performance on synthetic sequences

The new algorithm was run on a nine-frame sequence based on a small (50×50) image, **Figure 10a**, moving along the path shown in **Figure 10e**. The path was regularly sampled (in time) for this illustration and the grey-levels show the time taken to traverse each section of the path (the darker the pixel, the more time was spent traversing it). The motion template used was given perfect co-ordinates since

we are examining the response to image noise in the input sequence, not noise in the motion template. Again, a small image was chosen to make practical computation of large-scale tests. Noise was added at random to each frame of the sequence at eleven noise levels from 0% random coverage to 100% random coverage of the frame, with pixel values wrapping rather than being clipped when the addition of noise took the values out of range. The noise distribution was zero-mean Gaussian with a standard deviation of one. Examples of the effects of the increasing noise levels can be seen in **Figure 10**. The grey-level images produced are thresholded by the algorithm (shown before thresholding in **Figure 10b-d**). Note that the shape is completely obliterated at the maximum noise level and that at around 50% noise it is nearly impossible to distinguish the shape by human vision.

Again, the new technique is capable of coping with significant levels of noise. The performance curve in **Figure 11** is similar to those for previous VHT-derived techniques and, as mentioned above, earlier studies found that the VHT-based techniques are able to handle noise levels that are approximately twenty percent greater than a comparable GHT-based frame-by-frame technique.

3.2.2 Image-noise performance on real-world sequences

Our current intention is to apply the new techniques to the analysis of human motion and, eventually, for biometric use. For these purposes, and to substantiate the applicability to real-world imagery, we have evaluated the performance of motion templates when locating a walking person viewed from the side, as with the CVHT in section 3.1.3. Co-ordinates specifying the particular motion of the walker were gathered by selecting a reference point on the body and estimating its position by eye for each image in the sequence. Since these measurements are likely to contain inaccuracies, the motion template itself is not perfect and will be another source of errors and peak-spread in the accumulator. Again, as with the CVHT, owing to the robustness of its formulation there is no need for exceptional precautions in the new technique.

Figure 12a shows a reconstructed template of a walker's torso and head, which was created by manual tracing from one of the frames in the sequence. Also shown is a plot of the x and y components of the motion template used (**Figure 12e**) and several frames of the walker sequence CA1 with the extracted template superimposed (**Figure 12b-d**). A series of Gaussian noise tests showed performance similar to those

earlier for synthetic shapes, starting to lose accuracy at 40% noise, dropping off smoothly and missing the target completely above 80% noise. The difference can be attributed to the imperfect conditions prevalent in the real world, e.g. the cluttered and noisy background and the imperfect shape and motion templates.

A simple test of the effects of occlusion was carried out on the walker sequence CA1 described above. No noise was added to the sequence since this would be an unnecessary complicating factor. Instead, a number of vertical lines of pixels were blanked out and the algorithm was run on the resulting image. The results revealed that the new technique correctly extracts the walker until the blanking is 175 pixels wide – which completely obscures the walker for approximately 70% of the duration of the entire sequence. Furthermore, the extracted peak is within one pixel of the true peak for another thirty pixels, indicating there is a measure of peak spreading. As with other VHT derived algorithms, the new technique is capable of handling even extreme occlusion owing to the global integration of evidence across the entire sequence.

3.2.3 Effects of noise in the motion template

To determine the resilience to noise in the motion template, a percentage of co-ordinates calculated from it were perturbed by uniform noise. Note that adjusting these co-ordinates is equivalent to moving the target shape in the image sequence by the same amount. The strength of the noise determined the maximum distance (in pixels) that would be added to each co-ordinate. These tests were performed on the synthetic sequence and motion template in section 3.2.1.

The results show that as the percentage of co-ordinates corrupted increases, so the performance declines (**Figure 13a** and **Figure 13d**). This decline can be attributed to a "peak-spreading" effect common to all HT-derived techniques. As the accuracy of the input data decreases, the peaks in the accumulator become smaller and more spread out and the background noise level rises. To begin with, the HT-algorithm will find the correct parameters but, as noise increases and the definition of the correct peak becomes smoother, the parameters slip gradually from their true values. This continues until the spreading of the peak weakens it to the point that the algorithm is attracted to other potential sites. By comparing the graphs (**Figure 13b** and **Figure 13c**), it is clear that this is occurring. As the noise increases, the location of the peak moves in the neighbourhood by approximately the same amount as the pixel distance

noise being added to the motion template before finally switching to other incorrect peaks.

Only some portions of the motion template are significant - it is a continuous representation but the only significant points are those with a time reference matching the time references of the frames in the sequence analysed. This will have no effect on the validity of the results since uniform noise is applied. However, since it is entirely possible to have a high corruption rate without necessarily affecting the particular co-ordinates that are vital to a sequence, the graphs show a smooth decline.

The graphs all show a plateau effect where the maximum pixel distance is less than a particular amount (one pixel for the first two graphs, two pixels for the second two). This occurs when the noise is less than the "hit" threshold (again, one pixel for the first two graphs, etc). It is due to two effects: the discretisation process of accumulation rounds out the errors to within one pixel of accuracy and the post-processing for determining whether a hit has occurred flattens any other errors up to the hit threshold.

In summary, the motion template is sensitive to noise only when the points significant for the sequence being processed are affected by noise, and if their number is sufficient to overwhelm the in-built resilience of the evidence-gathering approach.

3.2.4 Finding people with motion templates

Further tests were performed to check the repeatability of human walker extraction. Three extra image sequences of walkers (MA1, SG3 and VH1) were extracted correctly as shown in **Figure 14**. Also shown are the results of attempting to extract the same walker from a different sequence (MA3, SG3 and VH3 respectively) whilst using the same shape and motion templates. As can be seen, MA3 and SG3 are correctly extracted but VH3 is slightly in error. This is due to some problems towards the end of the sequence with bad edges (and hence a bad motion template) resulting from height of the subject and the interaction with the lighting in the room.

Since the results are generally very accurate, this offers hope for a future attempt at locating humans with a "standard walk" motion template and a suitable range of scaling parameters. This has not yet been attempted due to the computational requirements but should be achievable with a genetic algorithm implementation.

4 Conclusions and Further Work

We have presented a new technique that robustly extracts optimal structural and motion parameters for arbitrarily moving arbitrary shapes in an image sequence. The technique requires no initialisation or training and has demonstrated excellent tolerance to noise and occlusion. Discretisation errors are minimised in the accumulator by using Fourier descriptors to represent the templates of shape and motion in continuous form, which eliminate common problems to do with rotation and scaling. Whilst the use of templates minimises effects of noise, exploiting the temporal correlation between frames maximises the possibility of correct extraction. The use of motion templates is a novel development for the HT and allows for a wide range of applications that require a more general motion model. This new capability comes without the explosion of parameter space dimensionality that would be inherent in current parametric approaches.

Future development will concentrate on two main areas. To further generalise the approach, it would be useful to partly relax the rigid shape requirement. A possible approach might allow the motion template to specify different models for different parts of the sequence, perhaps with some form of morphing between models loosely based on the HT for natural shapes [**18**]. In the application domain, we intend to integrate current gait research with motion templates. In combination with an articulated leg model, it may be possible to look for specific individuals using a motion template to describe a person's unique walking motion. This will require investigation of whether a motion template of a walking pattern is unique to an individual.

5 References

[1] J. K. Aggarwal and Q. Cai, Human Motion Analysis: A Review, CVIU, 73(3), pp. 428-440, 1999.

[2] I. A. Kakadiaris and D. Metaxas, Model based estimation of 3D human motion with occlusion based on active multi-viewpoint selection, Proc. CVPR, San Francisco, 1996, pp. 81-87

[3] J. E. Boyd and J. J. Little, Global versus structured interpretation of motion: Moving light displays, Proc. IEEE Computer Society Workshop on Motion of Non-Rigid and Articulated Object, Puerto Rico, 1997, pp. 18-25.

[4] V. F. Leavers, Which Hough Transform?, CVGIP: IU, 58, pp. 250-264, 1993.

[5] R. Polana and R. C. Nelson, Detection and Recognition of Periodic, Nonrigid Motion, International Journal of Computer Vision, 23(3), pp. 261-282, 1997.

[6] L. Xu and D. C. Hogg, Neural Networks in human motion tracking - An experimental study, Image and Vision Computing, 15, pp. 607-615, 1997.

[7] A. F. Bobick and A. D. Wilson, A State-Based Approach to the Representation and Recognition of Gesture, IEEE Trans. PAMI, 19(12), pp 1325-1337, 1997.

[8] J. M. Nash, J. N. Carter and M. S. Nixon, Dynamic Feature Extraction via the Velocity Hough Transform, Pattern Recognition Letters, 18(10), pp. 1035-1047, 1997.

[9] P. M. Merlin and D. J. Farber, A Parallel Mechanism for Detecting Curves in Pictures, IEEE Trans. on Computers, 24, pp. 96-98, 1975.

[10] D. H. Ballard, Generalising the Hough Transform to Find Arbitrary Shapes, CVGIP, 13, pp. 111-122, 1981.

[11] W.C.Y. Lam, K. S. Y. Yuen and D. N. K. Leung. Fourier Parameterization provides uniform bounded Hough Transform space, In: D. Chetverikov and W. G. Kropatsch, Eds., Lecture Notes in Computer Science 719 (5th Int. Conf. Computer Analysis of Images and Patterns '93, Budapest), Springer Verlag, Berlin/New York, pp. 183-190, 1993.

[12] A. S. Aguado, M. E. Montiel and M. S. Nixon, Bias Error Analysis of the Generalised Hough Transform, Journal of Mathematical Imaging and Vision, 12, pp. 25-42, 2000.

[13] F. P. Kuhl and C.R. Giardina, Elliptic Fourier Feature of a Closed Contour, CGIP, 18, pp. 236-258, 1982.

[14] A. S. Aguado, M. S. Nixon and M. E. Montiel, Parameterising Arbitrary Shapes via Fourier Descriptors for Evidence-Gathering Detection, CVIU, 69(2), pp. 202-221, 1998.

[15] J. M. Nash, J. N. Carter and M. S. Nixon, Extracting moving articulated objects by evidence gathering, Proc. BMVC 98, 2, pp. 609-618, Southampton, Sept. 1998.

[16] J. H. Han, L. T. Koczy, T. Poston, Fuzzy Hough transform, Pattern Recognition Letters, 15(7), pp.649-58, 1994.

[17] M. P. Murray, Gait as a total pattern of movement, American Journal of Physical Medicine, 46(1), pp. 290-332 1967.

[18] A. Samal and J. Edwards, Generalized Hough transform for natural shapes, Pattern Recognition Letters, 18, pp. 473-480, 1997.

Biographical Notes

Michael G. Grant, (BSc Computer Science) is a Research Assistant in the Image Speech and Intelligent Systems research group at Southampton. He is currently awaiting examination on his PhD. thesis on the topic of this paper. Mike is now working on the Automatic Gait Recognition project for the Human ID at a Distance program, funded by DARPA.

Mark S. Nixon is a Reader in the Image Speech and Intelligent Systems research group, and heads the vision research team. As well as research on feature extraction and description, applied to medical imagery and in biometrics (particularly gait recognition) he chaired the British Machine Vision Conference BMVC98, held at the University of Southampton, and has just completed a textbook forthcoming with Butterworth-Heinmann on feature extraction in computer vision. Mark is PI on the Automatic Gait Recognition program.

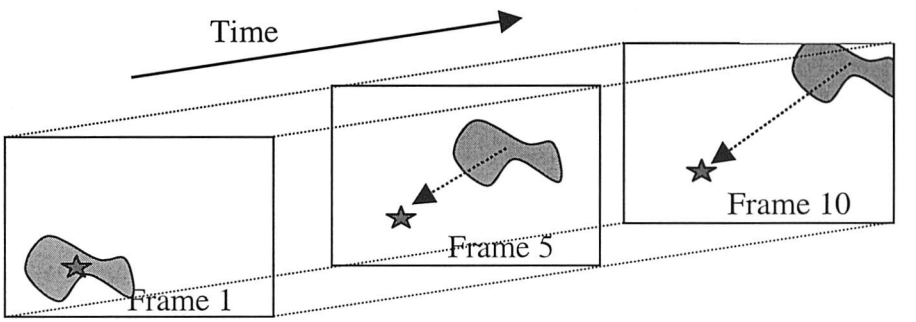
Paul H. Lewis is a Senior Lecturer in the Intelligent Agents and Multimedia research group. His main research interests are currently concerned with image and video analysis and their application to multimedia information handling. We are designing and developing novel facilities for multimedia information retrieval, navigation and browsing. The research involves work on content based image and video retrieval, hypermedia navigation, intelligent agents and an integrated semantic layer in the multimedia architecture.

(a) original shape (b) rotated and scaled versions of original shape

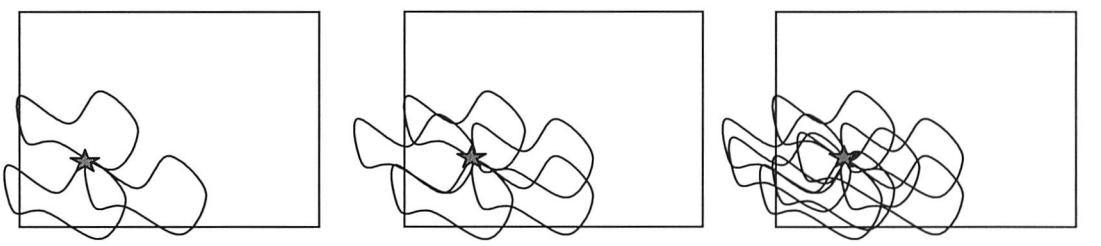
Figure 1 – Consequences of discrete shape representation

(a) first frame (0) (b) frame 10 (c) frame 18 (d) frame 26

Figure 2 - Space-Shuttle launch with extracted template (of the booster) superimposed in white



(a) simulated sequence of linearly moving arbitrary shape (star = start point)



(b) Accumulator after processing frame 1 (c) Accumulator after processing frame 5 (d) Accumulator after processing frame 10

Figure 3 – Simulated sequence and accumulator deriving from it

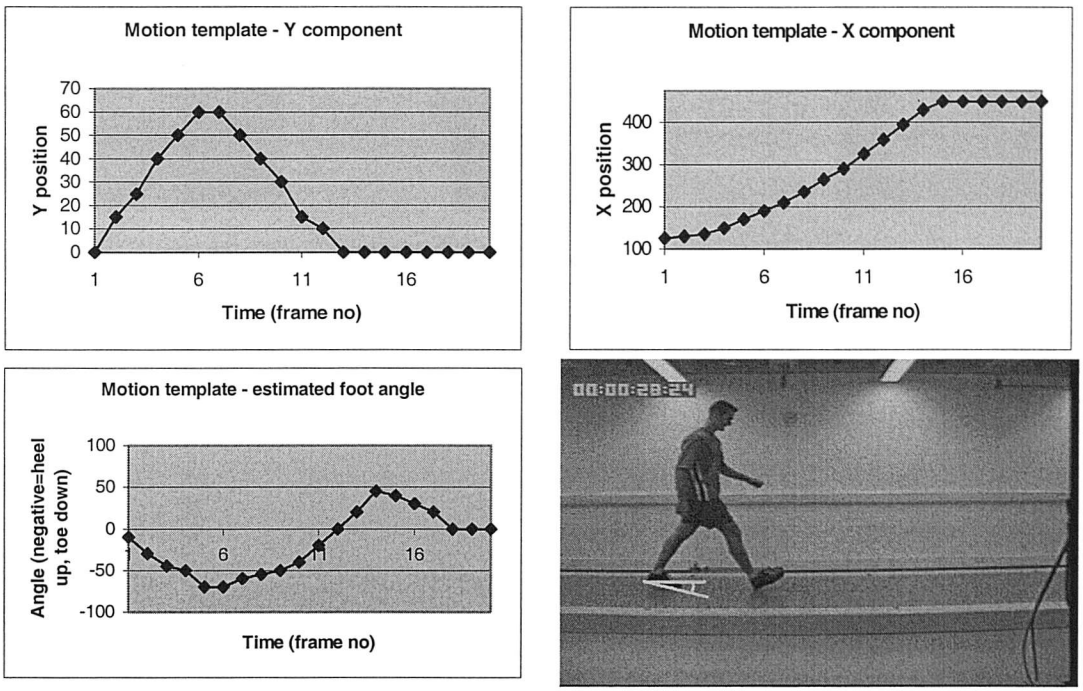
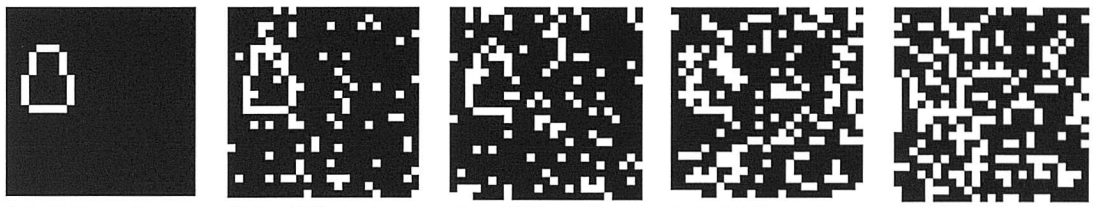


Figure 4 - Changes in x and y positions and rotation of a walker's foot angle



(a) 0% noise (b) 40% noise (c) 60% noise (d) 80% noise (e) 100% noise

Figure 5 – Artificial sequence with added Gaussian noise

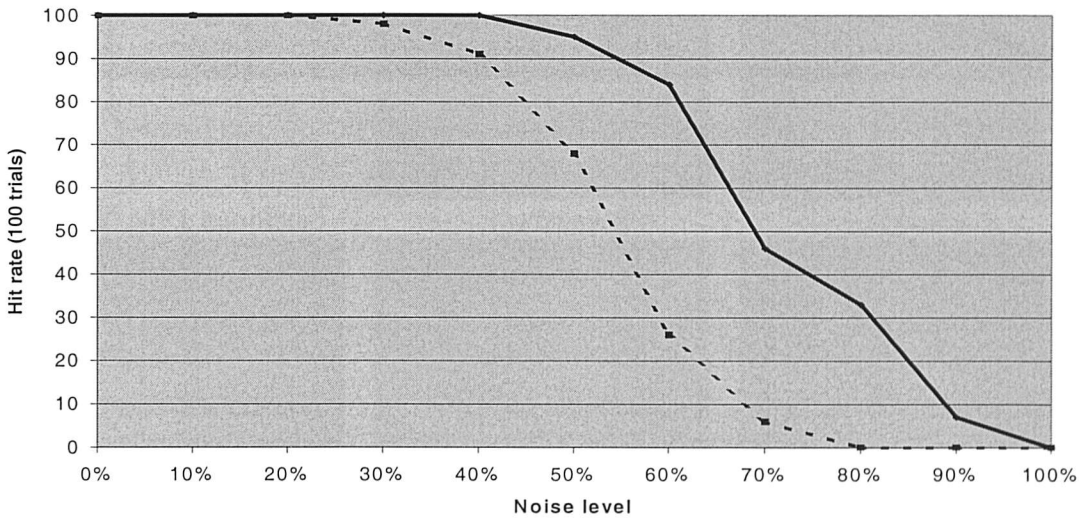


Figure 6 - Noise performance (dashed = GHT-based, solid = CVHT)

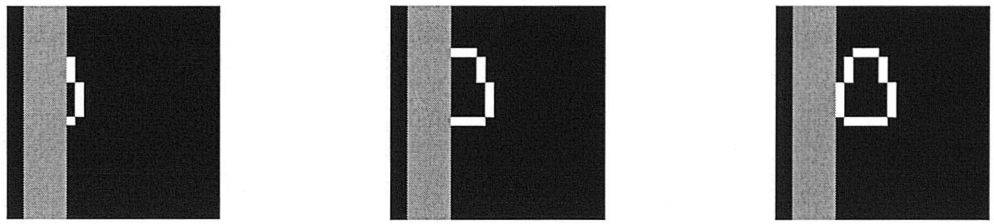


Figure 7 - Frames 1, 3 and 5 of occluded sequence (bar is 5 pixels wide)

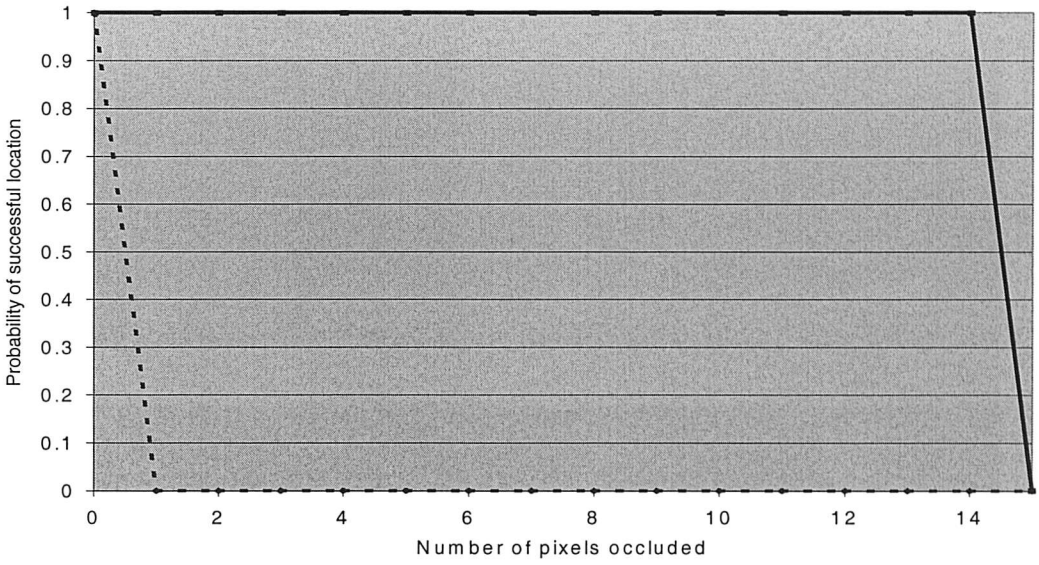
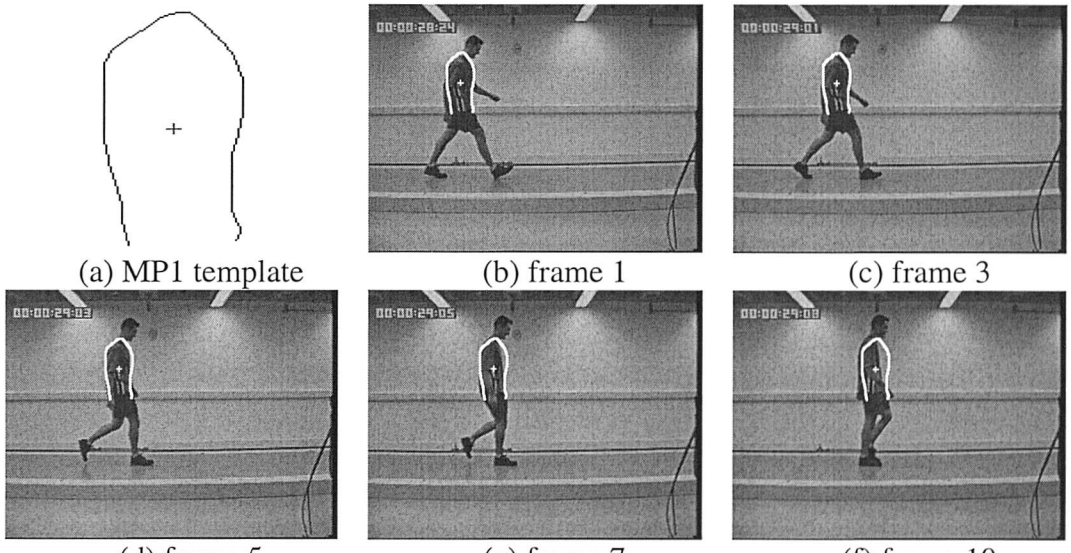


Figure 8 - Occlusion Tests (dashed = GHT-based, solid = CVHT)



(a) MP1 template (b) frame 1 (c) frame 3

(d) frame 5 (e) frame 7 (f) frame 10

Figure 9 - Extraction of MP1 sequence using a linear velocity model

(a) 0% noise (b) 20% noise (c) 50% noise (d) 100% noise (e) trajectory

Figure 10 – Original shape (with added Gaussian noise) and motion trajectory

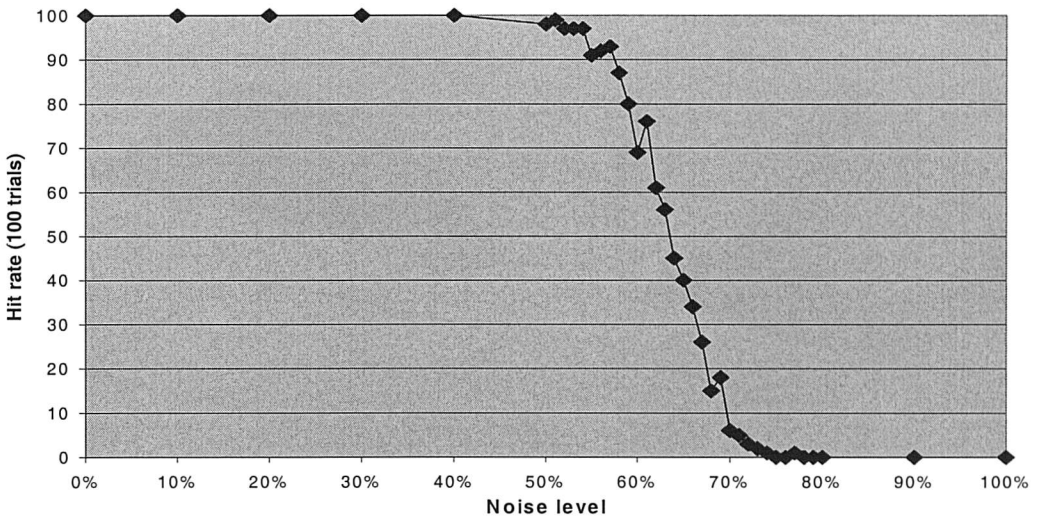
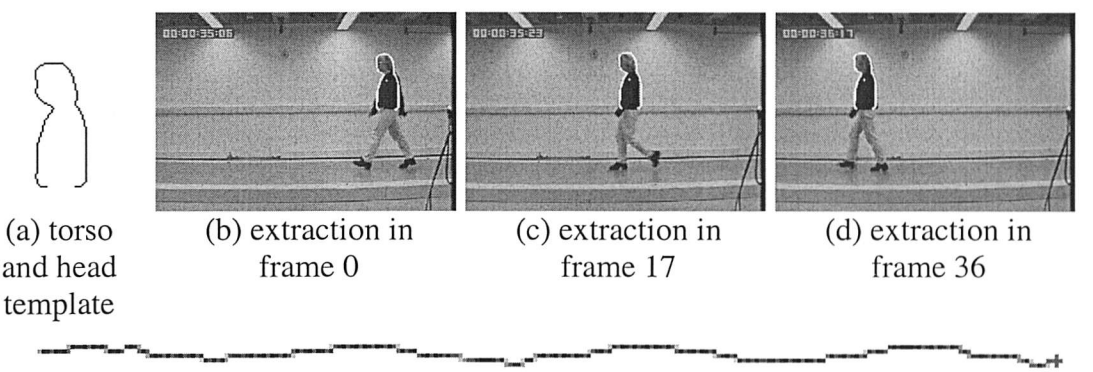


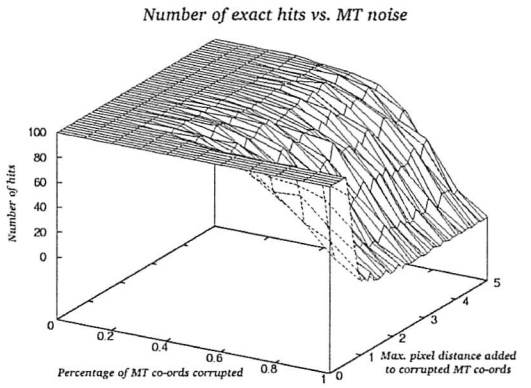
Figure 11 - Noise performance for synthetic imagery



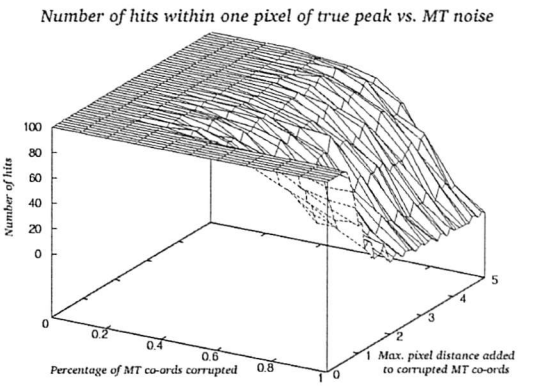
(a) torso and head template (b) extraction in frame 0 (c) extraction in frame 17 (d) extraction in frame 36

(e) motion template (begins at the small cross on the far right)

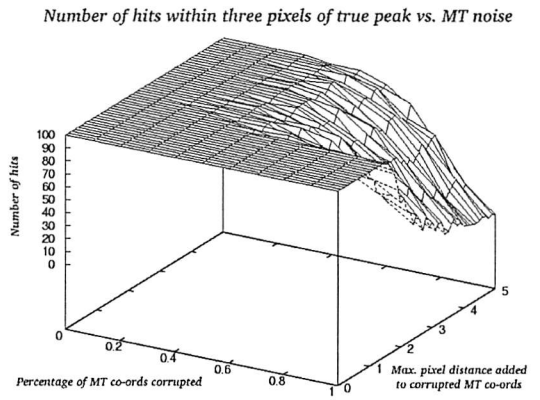
Figure 12 - Motion template and frames of sequence CA1 with superimposed template



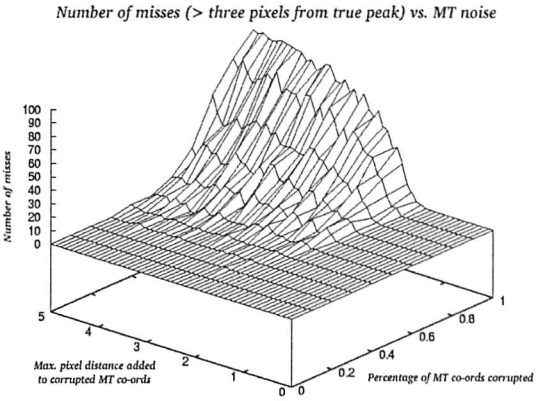
(a) exact hits



(b) near hits (within one pixel)



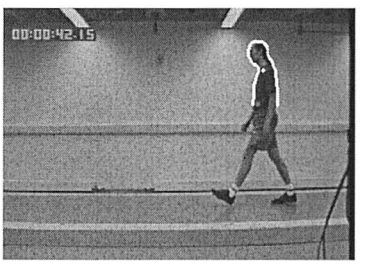
(c) near hits (within three pixels)



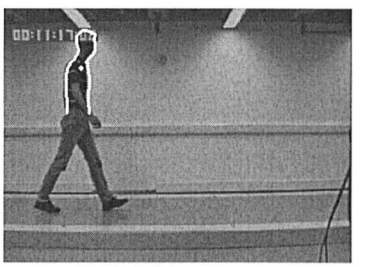
(d) total misses (different viewing angle)

Figure 13 - Hits and misses with percentage of corrupted co-ordinates in motion template

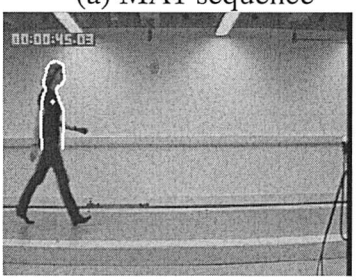
(a) MA1 sequence



(b) SG1 sequence

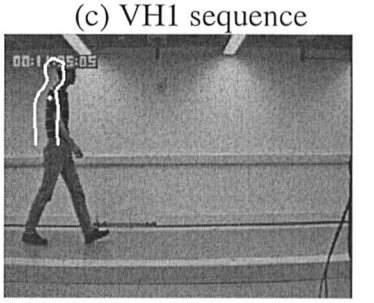


(c) VH1 sequence



(d) MA3 sequence

(e) SG3 sequence



(f) VH3 sequence

Figure 14 - Extracting different subjects