Chapter 7

7 Adding Temporal Constraints

7.1 Introduction

The deformation that has been 'learnt' thus far is time independent deformation. Models have been constructed that know **what** is valid deformation but not **when** deformation is valid. This important temporal constraint is beneficial in disambiguating models. When such mathematical constraints have been placed upon the deformation of an object in order to increase robustness, the important consideration of how a model moves with time should also be considered.

The linear formulation of the PDM makes iterative movements within the image frame based upon the assumption that the model will not alter considerably between consecutive frames. Providing a simple model and a slow moving/deforming object this assumption holds true. However, as has been demonstrated with non-linear models, this smooth iterative movement through shape space does not provide a sufficient mechanism to 'jump' between discontinuities in shape space. It is therefore apparent that if complex models are to be successfully tracked within the image frame, additional constraints must be applied to both increase robustness and to improve the transition through shape space.

The remainder of this chapter is concerned with the construction and use of temporal dynamics, which can be learnt in addition to deformation. Section 7.2 takes a graphical simulation example to construct a 3D non-linear PDM from which temporal dynamics are learnt. These dynamics can then be used to reproduce the deformation and motion of the model. Section 7.3 will discuss the issues of tracking complex non-linear models and how these temporal dynamics can be used to increase robustness and support multiple hypotheses. Section 7.4 demonstrates how these temporal constraints can be used to enhance classification. Lastly conclusions are drawn.

7.2 Learning Temporal Model Dynamics

7.2.1 Introduction

The work thus far has discussed the computer vision applications of non-linear models of shape and deformation, where models have been used to locate and track objects in the image frame. The models produce graphical representations of objects, which can be mapped to the appearance of real world objects within the image. In the field of computer graphics, similar representations are required for animation. The main difference is that graphical models are required to be 'life-like' and three-dimensional for rendering. The models must therefore exist in 3D. The rendering procedure then projects these models into 2D for viewing. In computer vision applications this projection is often incorporated into the statistical model, representing how an object deforms on the image-plane rather than within its own 3D co-ordinate system. However, this is not always the case and deformable models have also been applied to 3D in computer vision in order to reduce some of the non-linearity introduced during the projection process. [Heap 96; Ferryman 95; Hogg 83] have tackled computer vision from this 3D perspective, which is basically the reverse mapping of the rendering procedure. In computer graphics, [Pentland 96; Parker 97] have used statistics and interpolated models to produce 'life-like' renderings and animations of human facial motion.

The use of computer vision techniques in motion capture is common placein acquiring trajectories for key points of objects that are used to produce life-like 3D animations. Figure 7.2.1 and Figure 7.2.2 show motion trajectory files for a running and walking human female[9]. These were captured using reflective IR markers on a real world human subject. The trajectories of these markers in space were recorded in multiple camera views and the trajectories of these points calculated using standard stereo reconstruction techniques. The model consists of 32 3D-marker points and their trajectories through space. By connecting these points with a simple stick model the human motion can be visualized. In computer animation, these key points would be used to animate the articulated sections of a 3D virtual character for computer games or virtual environments.

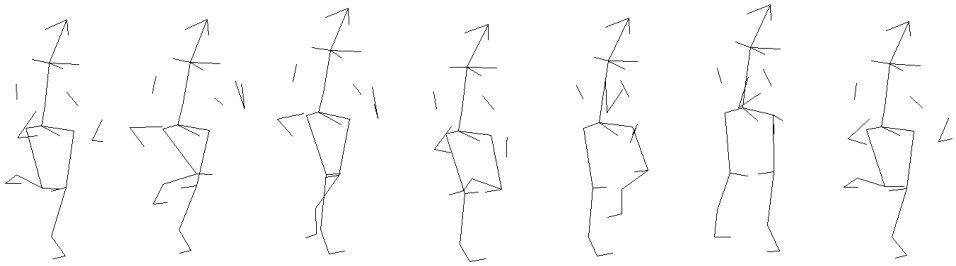


Figure 7.2.1 - Examples from a Key-frame animation of a Running Woman

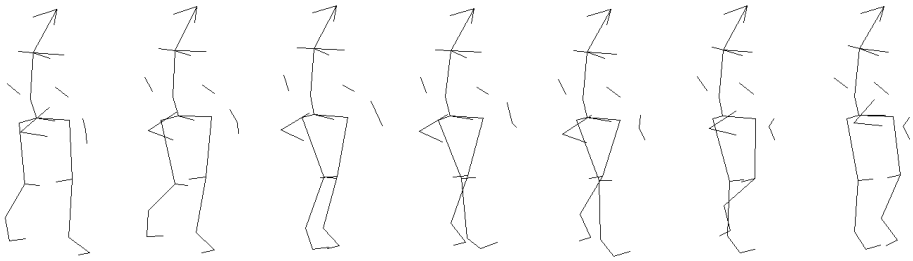


Figure 7.2.2 - Examples from a Key-frame animation of a Walking Woman

It is this notion of *key points* in the motion capture process that provides the link between statistical models and animation, where animation key points are akin to the landmark points used in statistical models. If statistical models of shape and deformation can be ***learnt*** from a training set, producing realistic constraints on the shape (or motion of landmark points), then similar ***learnt*** models of animation trajectories can also be achieved.

[9] The motion capture data for the female subject was provided by TeleVirtual Ltd.

7.2.2 The Linear Motion Model

The human motion capture data for both the running and walking woman consists of 32 key points for each frame of the animation; these points can be concatenated into a single 96 dimensional vector $V=(x_1,y_1,z_1, ..., x_{32}, y_{32}, z_{32})$. The running animation consists of 474 key frames recorded at 30Hz which produces a training set of 474, 96 dimensional vectors. The walking animation consists of 270 key frames, again captured at 30Hz using 32 key points producing a training set of 270, 96 dimensional vectors. Now the training sets are in a form that enables further statistical analysis: linear PCA can be performed upon them to produce a linear 3D PDM.

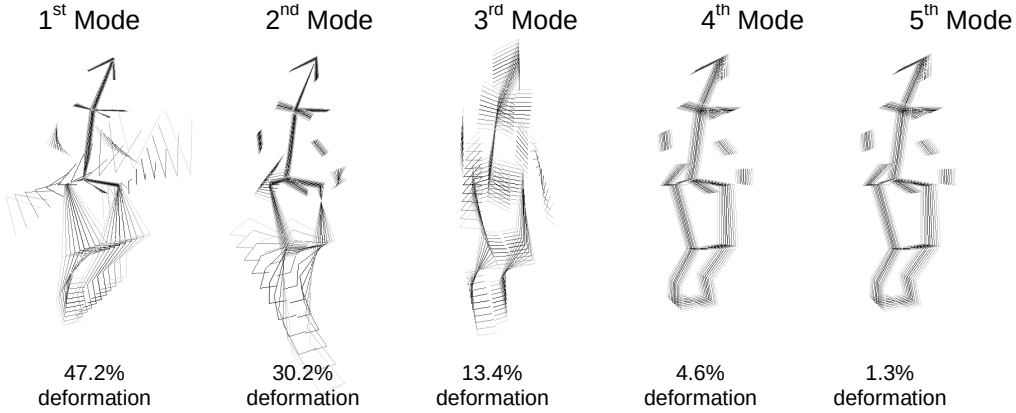


Figure 7.2.3- The Running Linear 3D PDM

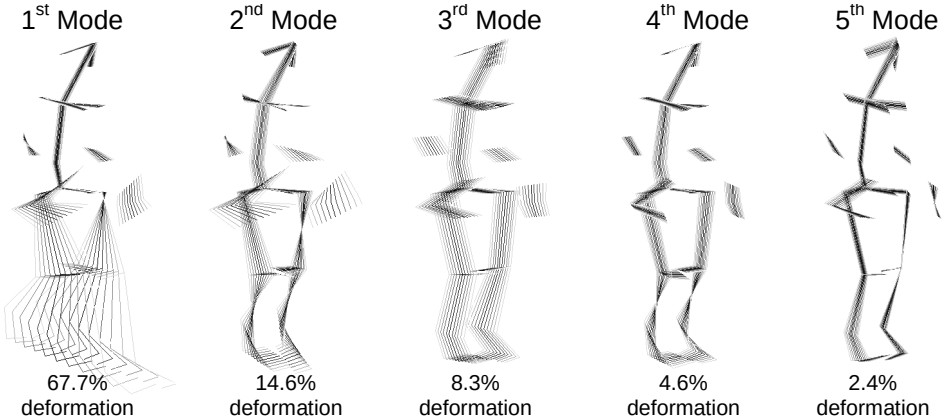


Figure 7.2.4 - The walking Linear 3D PDM

From the eigenvalue analysis, 98.8% of the deformation of the running model is contained within the first 10 eigenvectors, with 99.4% of the walking model being encompassed by the 10 eigenvectors.

It can be seen from Figure 7.2.3 and Figure 7.2.4 that the linear 3D PDM does not model the trajectories of key points (and associated body parts) well. The motion files contain perfect landmark point identification between examples. However, the data sets are still non-linear due to the circular motion of the body parts. This non-linearity can be seen in Figure 7.2.6 and will be discussed shortly. It should be noted that the 3rd mode of variation of the walking model encompasses mainly translation. This is due to the change in speed as the walker establishes a consistent gait, and remains a part of the model due to the absence of the alignment of the training examples. Had the normal alignment procedure been followed, then this translational information would have been reduced. The translation correlates to the shift in m_1 of the walking model seen in Figure 7.2.6b. However, this information is important to the realism of the animation and must therefore remain a component of the model. It will later be removed through the use of temporal dynamics.

7.2.3 Adding Non-linear Constraints

Using the methods previously discussed, the data sets are first dimensionally reduced by projecting each of the training examples down onto the eigenvectors of the linear PDM. Using the 10 primary modes of the linear model as determined in the previous section, both the running woman data and the walking model are projected down from 96 to 10 dimensions. These lower dimensional data sets are shown in Figure 7.2.6 as points drawn in 3D from two 2D views.

Cluster analysis was then performed on the reduced data sets. The resulting cost files are shown in Figure 7.2.5. The natural number of clusters for the run and walk trajectory files can be estimated to be 25 and 30 respectively. The larger number for the walking model is due to the model translation introduced as the subject establishes a consistent gait, as mentioned earlier.

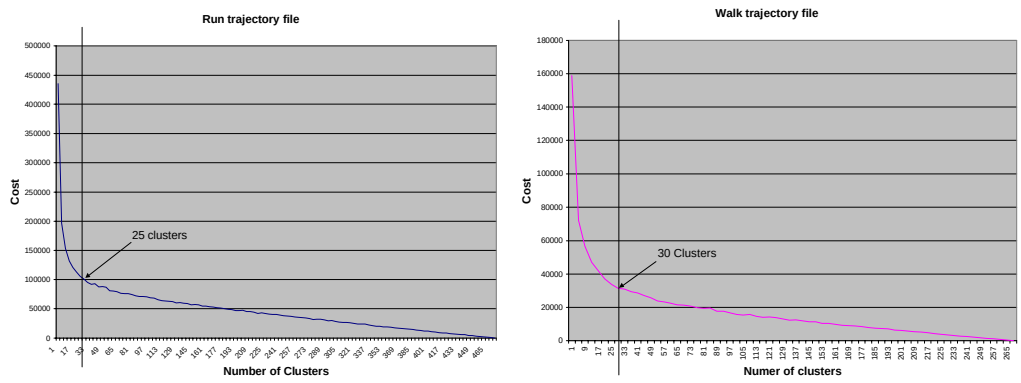
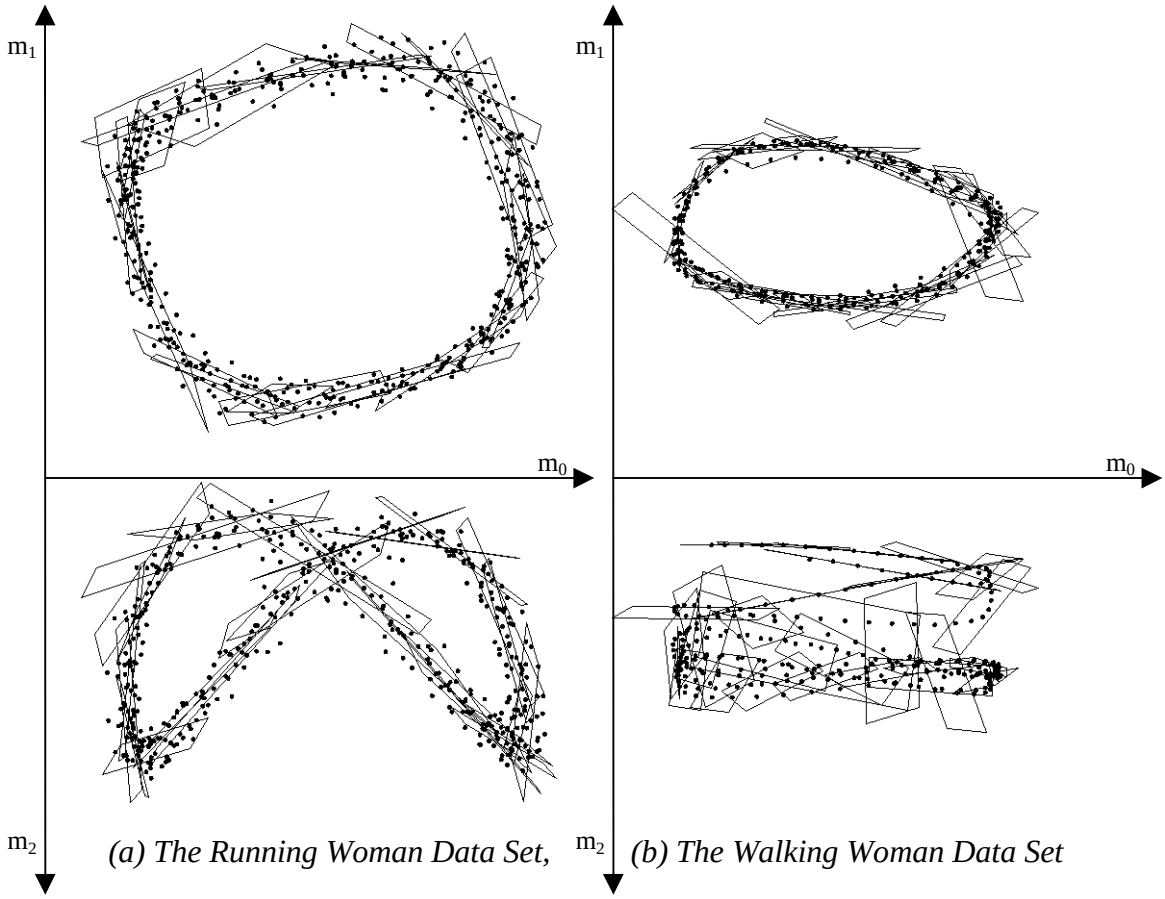


Figure 7.2.5 - Cost files for Trajectory Data

Using the natural number of clusters for each data set, the fuzzy k-means algorithm was used to segregate each data set into its composite clusters. Each cluster was then modelled by performing further PCA upon its members. The final non-linear constraints can be seen in Figure 7.2.6 with the bounds of each cluster drawn as a rectangle over the reduced data set.



(a) The Running Woman Data Set, *(b) The Walking Woman Data Set*

Figure 7.2.6 - Dimensionally Reduced Data sets with the Cluster Based Constraints

From this diagram it can be seen that the clustering algorithm has smoothly estimated the natural curvature of the data set through piecewise linear patches. Each cluster better estimates the model locally as each linear patch must encode less information.

The CSSPCA has ***learnt*** the *Motion Capture Space* and can be used to reproduce viable shapes from the model. However, in computer animation this is insufficient. For animation purposes, the ability to model the trajectory through shape space is also required, allowing the motion to be reproduced.

7.2.4 Learning Temporal Constraints

Thus far the techniques have been used to learn the shape and size of the trajectory space, temporal analysis must be performed to estimate how the model moves through space with respect to time.

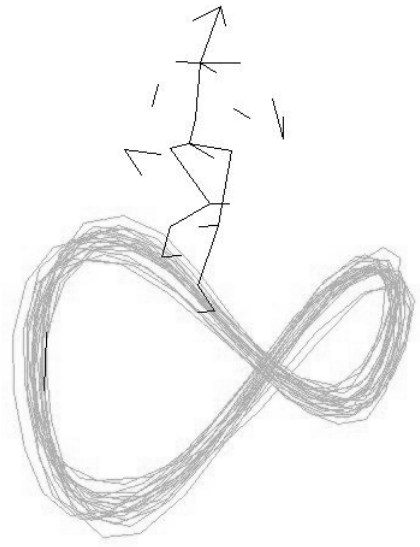


Figure 7.2.7 - Trajectory through Reduced Shape Space

Figure 7.2.7 shows the 3D trajectory of the reduced dimensional running data set projected down into 3 dimensions. Using simple animation techniques it is possible to watch the model move throughout the space as the animation sequence iterates. It is apparent that the motion is cyclic and consistent in nature and repeats in accordance with the period of the stride of the actor. Therefore,

given any point within the space it is possible to predict where the model will move to next, based upon this observed motion.

The model has been estimated in a lower dimensional space; if the trajectory can also be modelled in this lower space then it is likely that paths of motion throughout the space could be determined and reconstructed. The key again is in this probabilistic analysis of the training set. The deformation constraints have already broken the shape space down into linear patches with the centre of the clusters being the mean shape of the transition at that point in time. It is also known that, due to the cyclic nature of the data set, the pattern of movement repeats at regular intervals for fixed speeds of motion. Although this is not a necessary condition, it can effectively be modelled as a self-starting, finite state machine. This lends itself naturally to a discrete, time dependent, probabilistic analysis of the motion.

The reduced training set can therefore be used to analyse the model and probabilistically learn the transition of the model between clusters. This can be done with a state transition matrix of conditional probabilities, otherwise known as a Markov chain.

7.2.5 Modelling Temporal Constraints as a Markov Chain

A Markovian assumption presumes that the present state of a system (S_t) can always be predicted given the previous n states (S_{t-1}, S_{t-2}, ..., S_{t-n}). A Markov process is a process which moves from state to state dependent only on the previous n states. The process is called an order n model where n is the number of states affecting the choice of the next state. The simplest Markov process is a first order process, where the choice of state is made purely upon the basis of the previous state. This likelihood of one state following another can be expressed as a conditional probability $P(S_t|S_{t-1})$.

A Markov analysis looks at a sequence of events, and analyses the tendency of one event to follow another. Using this analysis, a new sequence of random but related events can be produced which have properties similar to the original.

The probability mass function $P\left(C_j^{t_n}\right)$ denotes the unconditional probability of being in cluster *j* at time t_n, or being in state *j* after *n* transitions (time steps). A special situation exists for *n=0* where $P\left(C_j^0\right)$ denotes the probability of starting in state *j*. However, due to the assumption that the motion is cyclic and the trajectory file starts and ends mid-cycle, no information is available for these initial probabilities.

The conditional probability mass function is therefore defined as

$$P\left(C_j^{t_n}\middle|C_k^{t_m}\right)$$

$P\left(C_j^{t_n}\middle|C_k^{t_m}\right)$ gives the probability of being in cluster *j* at time t_n conditional on being in cluster *k* at time t_m. In the trajectory file example it is fair to make the assumption that the next state of the model can be determined from the previous state. This can be confirmed by observing the trajectory taken through shape space by the training set (see Figure 7.2.7). Provided stationary elements of the chain are ignored, i.e. where $P\left(C_j^t\middle|C_j^{t-1}\right)\geq \max_k\left(P\left(C_j^t\middle|C_k^{t-1}\right)\right)$ and therefore choosing the 2nd highest probability move at each time step, the continuous transition through shape space can be achieved. If this assumption is made, then the process becomes a first order *Markov process* or *Markov Chain* and $p_{j,k}$ a one step transition probability

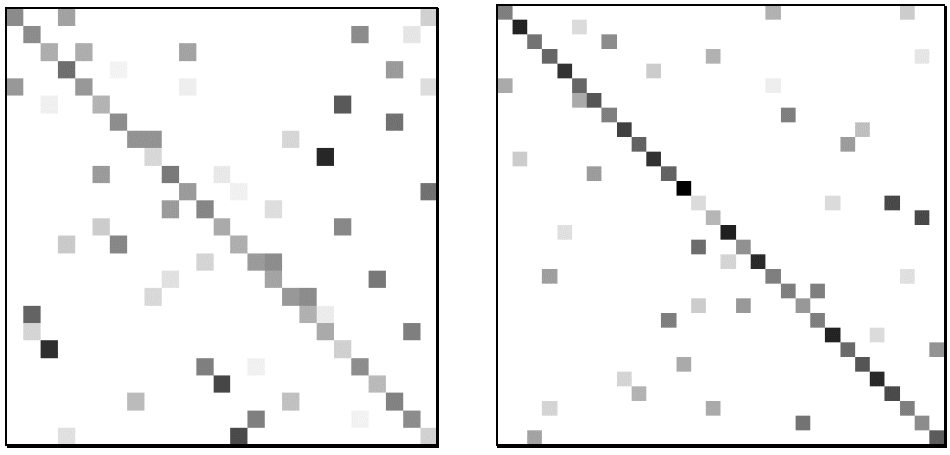
$$p_{j,k} = P\left(C_j^t\middle|C_k^{t-1}\right)$$

If there are *n* clusters in the model, then there are *n* states in the chain, hence a state transition matrix is an $n\times n$ matrix of one step transition probabilities. This is constructed in a similar manner to the classification probability matrix constructed in section 6.5.6, and is a discrete probability density function (PDF).

$$P\left(C_j^t \middle| C_k^{t-1}\right) = \begin{pmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,k} \\ p_{2,1} & p_{2,2} & & \vdots \\ \vdots & & \ddots & \vdots \\ p_{j,1} & \cdots & \cdots & p_{j,k} \end{pmatrix},$$

where $p_{j,k} \geq 0$ for all *j,k*, and $\sum_k p_{j,k} = 1$ for all *j*.

After construction of the PDF its content can be visualised by converting the matrix to a grey-scale image. Figure 7.2.8 shows the resulting images for both the running and walking data sets. It can clearly be seen that high probabilities exist along the diagonal of the image. This diagonal, when $i=j$ or $S_t=S_{t-1}$, demonstrates that the model always has a high probability that it will stay within the same local patch. This can be attributed to the discrete nature of the model, and the fact that each patch is constructed to model local deformation. The darker diagonal in the walking model shows that this model has a higher probability of remaining within a local patch and is a result of the speed of movement. As both sequences were captured at the same rate, the slower movement of the walking model generates more frames in each local patch and hence a lower probability that the model will make a transition to another patch. However, as the numerical identity of each local patch within the matrix is randomly generated by the k-means algorithm, no further conclusions can be drawn from the patterns within the image, hence the random distribution.



(a) The Running Woman Data Set *(b) The Walking Woman Data Set*

Figure 7.2.8 - Discrete Probability Density Functions

The *PDF's* shown in Figure 7.2.8 provide a conditional probability that, given a cluster at time *t*, the system will move to another cluster at the next time step. By taking the highest probability move at each time step the highest probability path can be modelled throughout the space.

Using this information and the mean shape of each cluster as key frames, the motion of the training set can be reconstructed. If any cluster of the model is chosen at random and the next highest probabilistic transition made at each time step $\mathbf{argmax}_i\left(p_{i,j}\right)$ where $i \neq j$, the model should settle within a natural path through the space. This is similar to a finite state machine that has a circular path and is self starting. If the natural number of clusters selected is correct then the cyclic period of the model should be equal to that of the training set. If the cluster number is too high then non-equidistant cluster centres result and the model appears to 'jerk'. If the cluster number is greater than twice the natural number then the model risks having a cyclic period of multiples of that of the true motion.

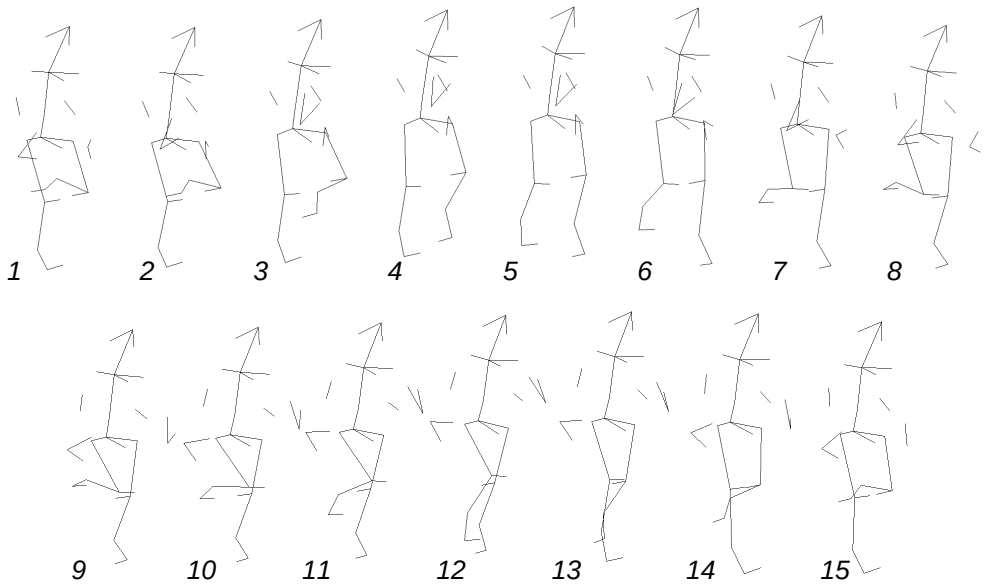


***Figure 7.2.9* - Extracted Trajectory for Running Model**

Figure 7.2.9 shows the highest probability path for the running model that consists of 15 clusters. Each pose of the model is the mean shape (exemplar) of a cluster. This model is reconstructed from the information that has been learnt

from the motion file and accurately reproduces the original motion. The animation can be further refined by linearly interpolating between these key frames (exemplars), as the linear interpolant along a line between exemplars is equivalent to linearly interpolating all points on the model between key frames. This does however introduce slight non-linear deformities. These deformities can be reduced by projecting the interpolated model into the constrained space to extract the closest allowable model for rendering.

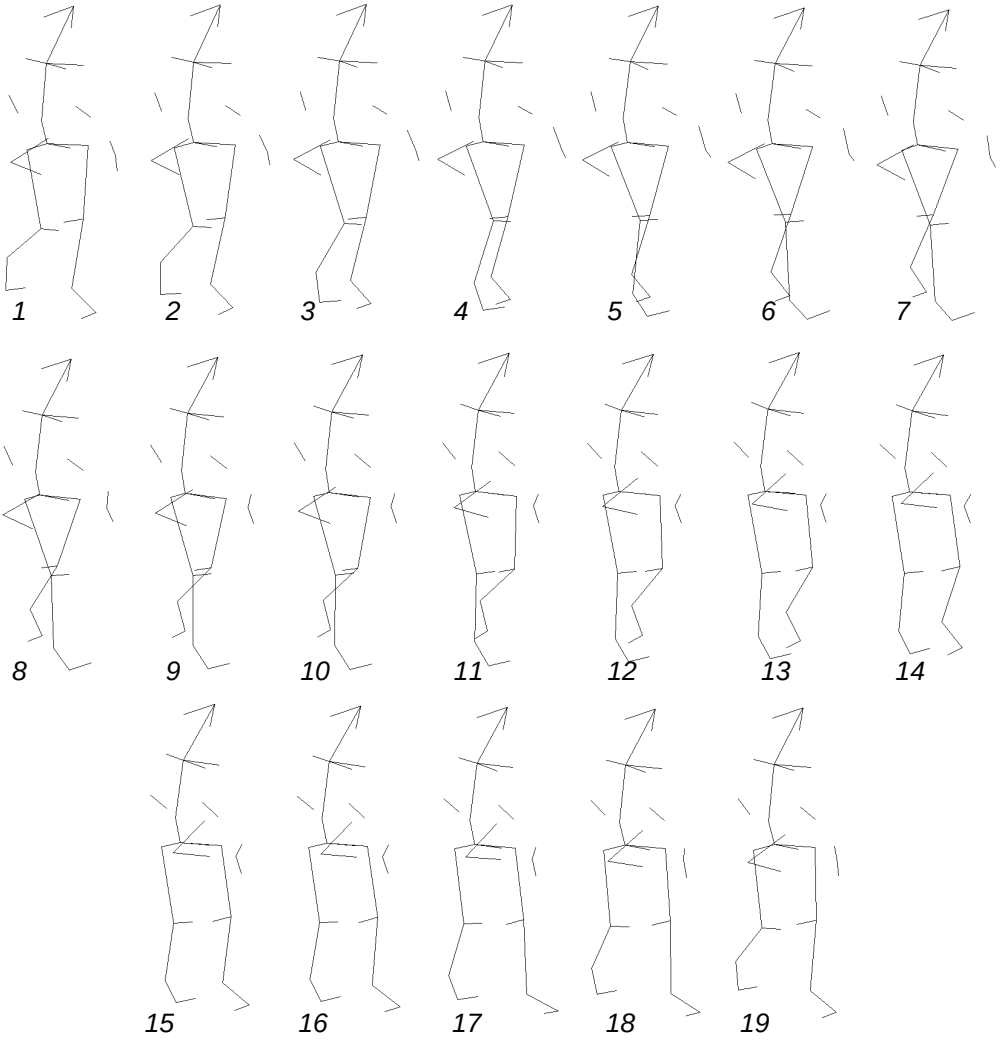


Figure 7.2.10 - Extracted Trajectory for Walking Model

Figure 7.2.10 shows the highest probability path through the walking model, consisting of 19 key frames that produce a cyclic path of high probability through the Markov chain. The original model contained 30 clusters and the redundant 11 clusters partly model the introductory gait acceleration, which can

be seen in Figure 7.2.11. The red line shows this high probability path extracted from the Markov chain. Acting like a self-starting finite state machine, if the model is initiated within the low probability startup area of the space, the chain quickly moves the model to the circular region, where constant cyclic movement occurs.

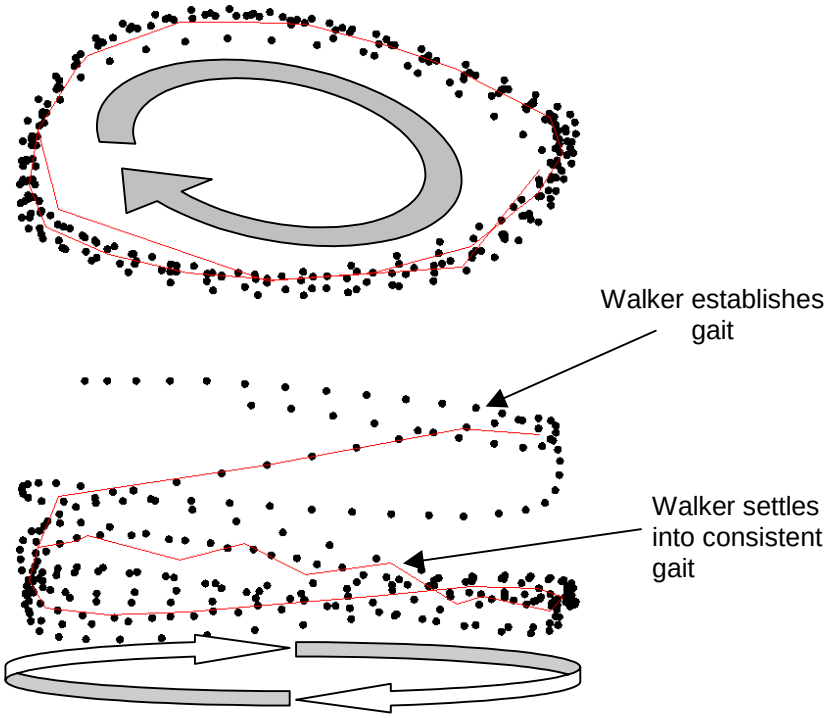


Figure 7.2.11 - High Probability Path through Walking Model Shape Space

7.2.6 Conclusions

In this section it has been shown how the reduced dimensionality and discrete representation of the Constrained Shape Space approach to modeling non-linear data sets can be used to provide simple analysis and reconstruction of motion. This is done by analysing the training set and constructing a Markov Chain, which is a discrete, probabilistic representation of the movement of the model through shape space. It has also been shown how, using this learnt temporal information, animated models can be produced which encapsulate the temporal information learnt from a training set.

7.3 Tracking with Temporal Dynamics

7.3.1 Introduction

In the previous section, temporal information was learnt from a training set in addition to deformation. It has been shown how this temporal deformation can be used to represent and reproduce motion. However, for many computer vision techniques this is not the ultimate goal. What is beneficial is using this *learnt* temporal information to further constrain the model, or predict the movement and deformation of an object, thus producing more robust tracking and classification.

A large body of work has been performed on the temporal mechanics of tracking. Many researchers have attempted to use predictive methods such as those based within a Kalman filter framework [Blake 98]. Hill *et al* proposed using genetic algorithms to model the discontinuous changes in shape space/model parameters [Hill 91][Hill 92].

Of particular interest to the work presented in this thesis is the CONDENSATION algorithm [Isard 98] [Blake 98] which is a method for stochastic tracking where a population of model hypotheses are generated at each iteration. These populations are generated from pre-learnt PDFs generated over the model parameter space to provide a hypothosis-and-test approach to model prediction and tracking. A more comprehensive introduction to Condensation is given in Section 2.5.

Condensation is a powerful tool in deformable model tracking for several reasons:

1. It supports multiple hypotheses and therefore produces robust results for tracking with occlusion and discontinuous movement.
2. It uses *a priori* knowledge about the object to predict its movement.
3. It recovers well from failure, allowing the model to 'jump' out of local maxima/minima.

It has been shown that, due to the discrete nature of the piecewise linear approach to modeling non-linearity, the approach directly lends itself to a discrete PDF with the addition of the Markovian assumption.

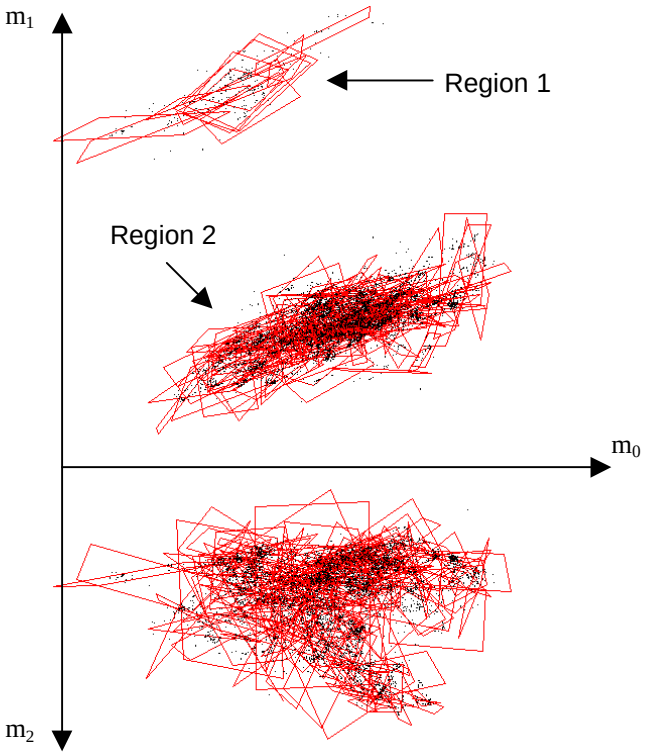


Figure 7.3.1 - Constrains on PCA space for the ASL Model

This temporal information can be used to augment the CSSPDM model with conditional probabilities, which allow the support of multiple hypotheses similar to that used in Condensation. This is important due to the discrete nature of the piecewise linear model. If the discontinuous shape space constructed for the American Sign Language (ASL) alphabet is considered from Section 6.5.6 (see Figure 7.3.1), it can be seen that shape space is segregated into **at least** two separate regions due to the movement of landmark points around the boundary (see section 2.4 for a description of these types of non-linearity). Furthermore, connected patches of the model may not represent consistent movement of the model in the image frame. This leads to the model *jumping* between patches, even when within region 2. Under these circumstances it is not possible for the

iterative refinement algorithm used for the classic PDM/ASM (section 3.3) to provide the *'jump'* between regions.

An image sequence was recorded of a hand signing the word 'gesture' which consisted of 170 frames. Figure 7.3.2 shows the model attempting to track the image sequence for the letters 'e' and 'u'. The model successfully tracks the letter 'e' but when the image sequence reaches the letter 'u' and the fingers elongate, the model is unable to make the jump to the new cluster responsible for modeling this letter. This problem is fundamental to the operation of the least squares iterative refinement algorithm and is due to two reasons:

1. Only a small section of the contour (marked in frame 'u') is responsible for 'pulling' the contour up to follow the elongated fingers. As this section is relatively small, compared to the remainder of the contour, it has less influence over the overall movement.

2. The maximum movement of the contour per iteration is governed by the length of the normal used to search around the contour. Hence this factor limits the distance the model can move through shape space at each iteration.

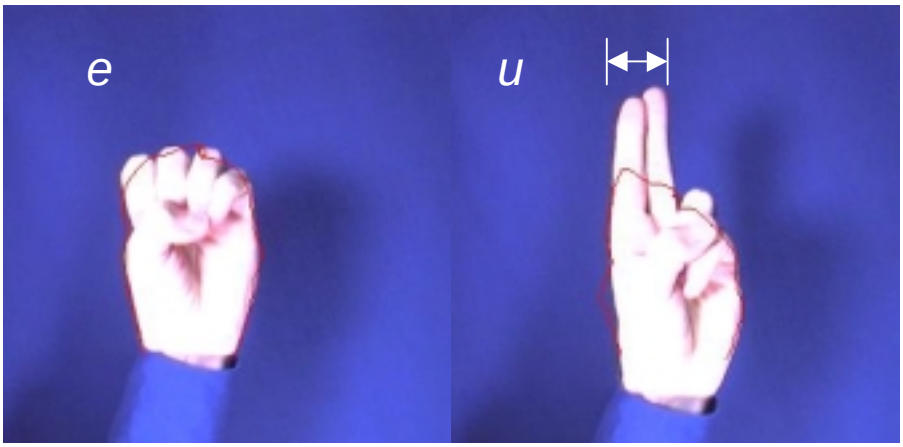


Figure 7.3.2 - ASL model Tracking an Image Sequence of the word 'gesture'

An obvious solution to these problems is to increase the search length along normals. Figure 7.3.3 shows the results of various parameters for the least squares iterative refinement algorithm on the ASL model. The graph

demonstrates the effect of varying the number of iterations per frame and the length of the normal (in pixels) either side of the contour. The cost at each iteration is the sum of the pixel difference between the desired movement of the model (gained from the assessment of the normals) and the final shape (after the constraints of the model have been applied). Where multiple iterations per frame were performed, these are displayed as fractions of a frame to visualise the resulting error cost of iteration. The corresponding letters of the sequence are shown with the vertical lines denoting the approximate transition between letters. At these transitional frames, the model error rises due to the increased speed of movement of the hand. During these faster movements the iterative refinement procedure must make larger movements through shape space to deform with the image. This produces the increase in error due to the limiting factor of the localised normal search.

Increasing the number of iterations produces a resulting reduction in cost up to a certain threshold, at which point the cost begins to rise again. This can be attributed to the finer iterations allowing the model to achieve poses from which it can not easily extract itself and is a further drawback of using the least squares iterative refinement approach to fitting a non-linear model. Although the increased normal length allows the model to achieve the aforementioned transition to the letter 'u', the resulting cost demonstrates a reduction in the overall performance of the model. The larger normal search allows the contour to affix to incorrect features in the image and hence results in degradation. Where image sequences with heavy background clutter are considered, this problem becomes more acute.

Another drawback of large normal searches is the resulting computational cost in assessing the additional pixel intensity gradients. It is therefore necessary to use a tracking paradigm that allows these quantum leaps in shape space to be made while retaining the localised searching and constraints of the model.

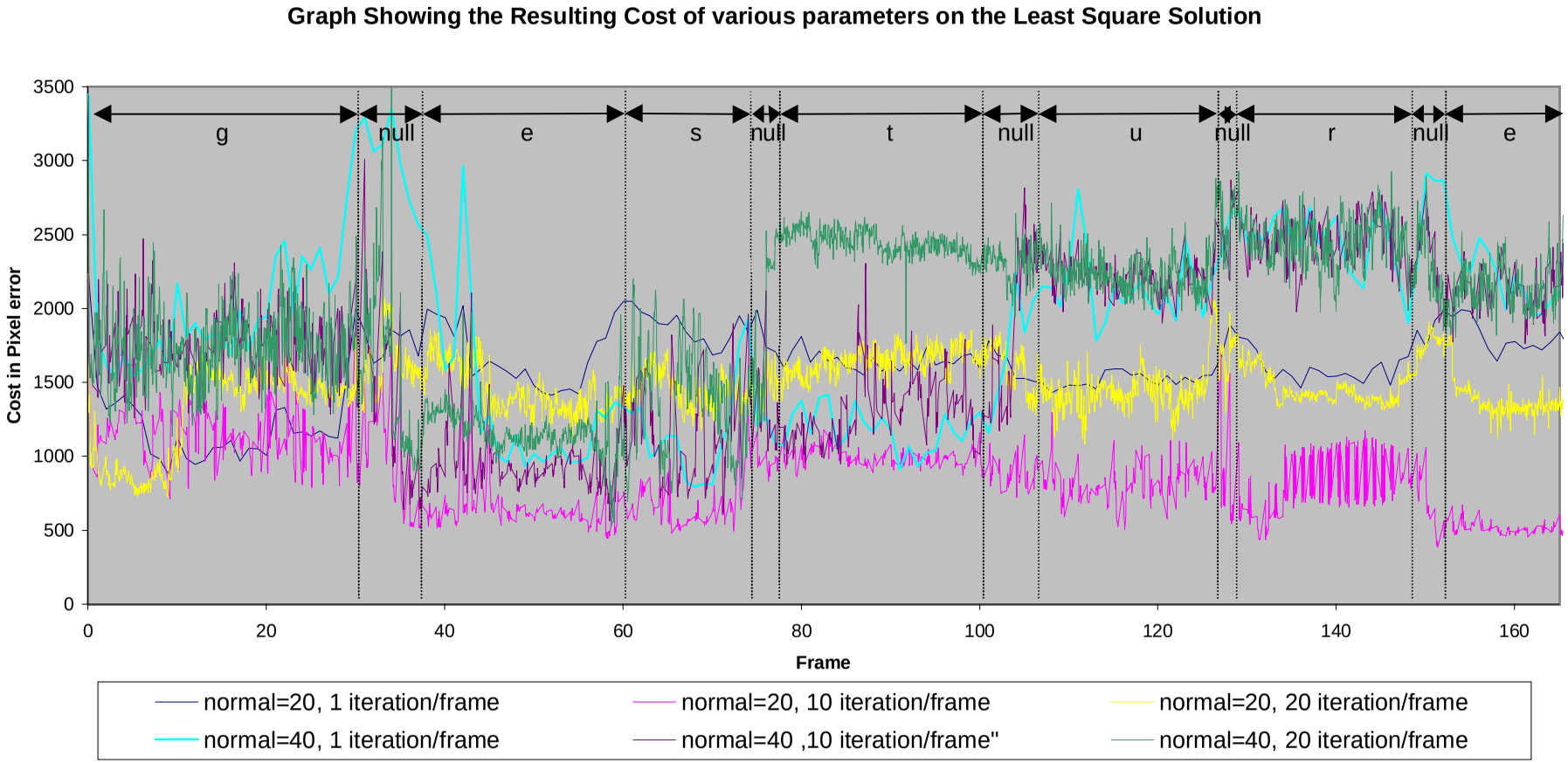


Figure 7.3.3 - Graph of error cost for Least Squares Fitting with Various Parameters

7.3.2 Finding the Optimal Ground Truth for Tracking

To locate the optimum solution (i.e. the closest allowable shape from the Constrained Shape Space PDM, CSSPDM) for each iteration of the model, the space was exhaustively searched. If the assumption is made that any local patch of the CSSPDM can indeed be treated as a linear model, then the iterative refinement procedure can be used to move locally within that patch to the closest possible shape. Therefore, if the best match within each patch (cluster) is located for each frame, the resulting lowest cost solution must be the (near) optimum.

This exhaustive search was performed on the 'gesture' image sequence. For every frame, each of the 150 clusters were assessed in turn. The mean shape of the cluster was used as a starting shape and the iterative refinement of the model, within the cluster, performed until the model converged (typically 40 iterations). The cluster that produced the lowest cost solution was deemed to be the optimum and the resulting costs plotted in Figure 7.3.4 along with the lowest of the least squares approaches from Figure 7.3.3.

The two smoothed plots are polynomial trendlines fitted to the data to help visualise the overall efficiency of the approaches. The optimum solution produces a lower error than that of iterative refinement, which would be expected. However, both exhibit similar trends. From this it can be inferred that some of the errors produced during tracking are not the result of the algorithm's inability to track successfully but are due to the constraints of the model. The higher error rates that result from letters such as 'g' and 'r' suggest that more training examples for these letters are required so as to increase the ability to model unseen shapes.

By analysing the optimum path through shape space and comparing this with the path taken by the least squares approach, the notion of discontinuity within shape shape can be confirmed.

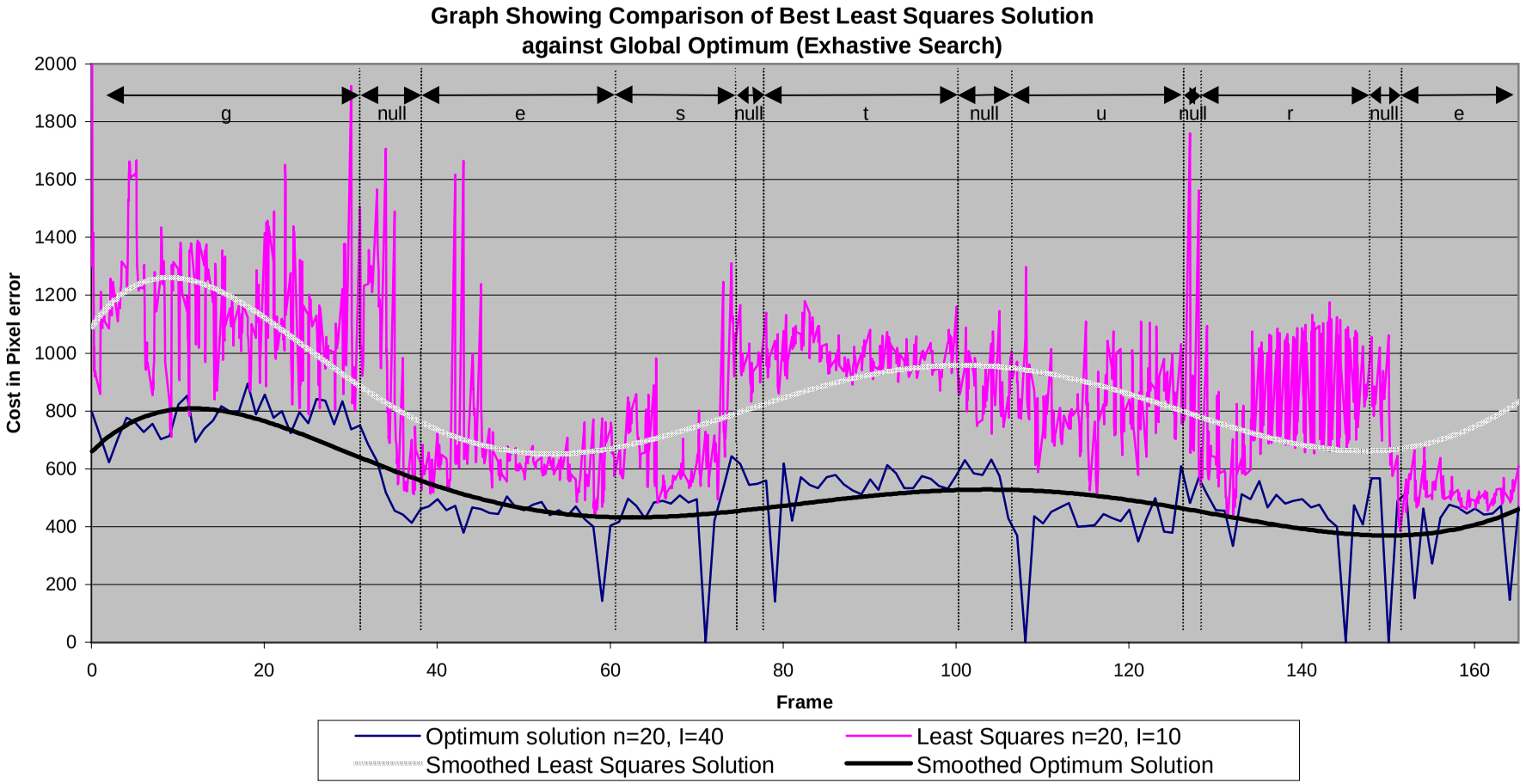


Figure 7.3.4 - Comparison of Least Squares Solution against Optimum Solution

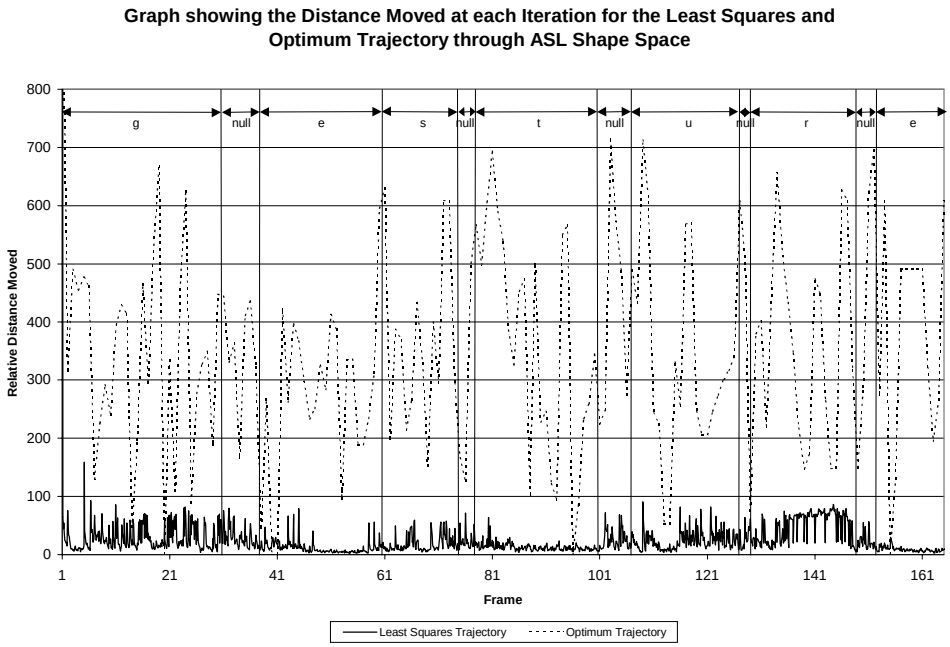


Figure 7.3.5 - Graph of Distance Moved at each iteration for Least Squares Solution and Optimum Solution

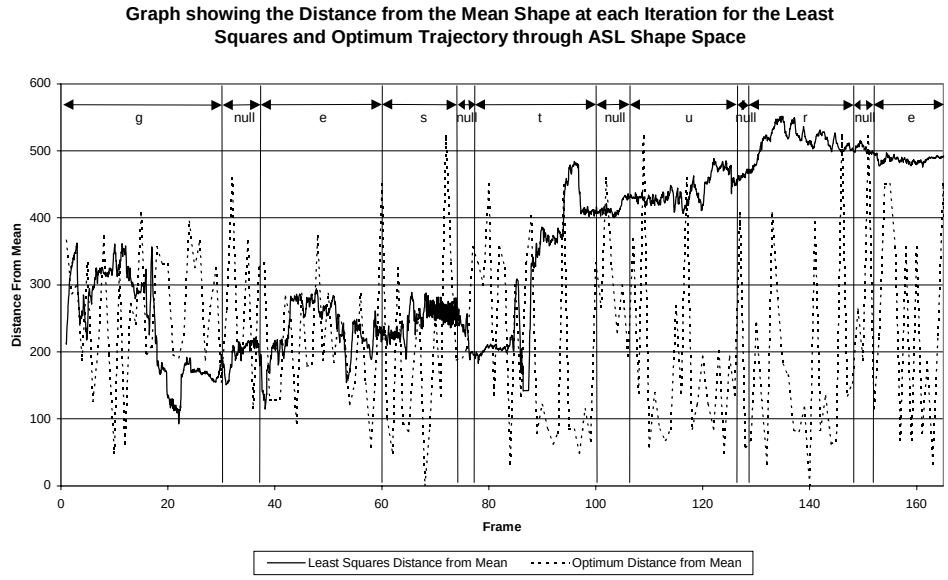


Figure 7.3.6 - Graph of Distance from Mean of Shape Space at each frame for Least Squares Solution and Optimum Solution

Figure 7.3.5 shows the distance moved through shape space at each iteration for both the optimum trajectory and the iterative refinement algorithm. From this it can clearly be seen that the least squares iterative refinement algorithm makes small incremental movements at each iteration, whereas the optimum trajectory

makes large 'jumps' at every frame. During the letters 'e' and 't' the least squares approach almost stops moving, which demonstrates that the model has converged upon a stable solution. However, the lack of such trends for other letters shows that the model is constantly struggling to better refine itself. Figure 7.3.6 shows distance from the centre of shape space for the two trajectories at each iteration. Again this demonstrates that the optimum path jumps violently within the space whereas the least squares approach makes small movements. The high values achieved by the least squares approach for the letters 'u' to 'e' show that the model is at the extremity of shape space making small movements. However, the relative movement of the model in Figure 7.3.5 for frames 100-150 show that it is moving considerably at each iteration attempting to find a better solution.

The most interesting aspect of these figures is within Figure 7.3.6. The letter 'e' occurs twice during the sequence. However, during the first occurrence the least squares approach is at a distance of around 200 units from the mean whereas during the second occurrence it is at around 500. This demonstrates two facts:

1. That there are at least two areas of shape space responsible for modeling the letter 'e' and these are distinctly separated in shape space.
2. The least squares approach can only use the local 'e' part of shape space and is incapable of jumping between them.

This confirms that not only is the non-linear shape space discontinuous but the least squares iterative refinement approach is incapable of providing a robust method for tracking. Instead a new method of applying CSSPDMs must be devised.

7.3.3 Supporting Multiple Hypotheses

By taking advantage of the Markovian assumption, a similar model of temporal dynamics can be generated for the ASL model as was constructed for the motion capture data previously discussed, where the conditional probability $P\left(C_i^{t+1} \middle| C_j^t\right)$ is calculated. As has been discussed, the major discontinuities of the shape space occur when landmark points jump around the boundary and hence result in a

jump in shape space (Figure 7.3.5 and Figure 7.3.6). However, within each patch, the model still makes small iterative movements. This can be confirmed by visualising the resulting PDF as a grey scale image.

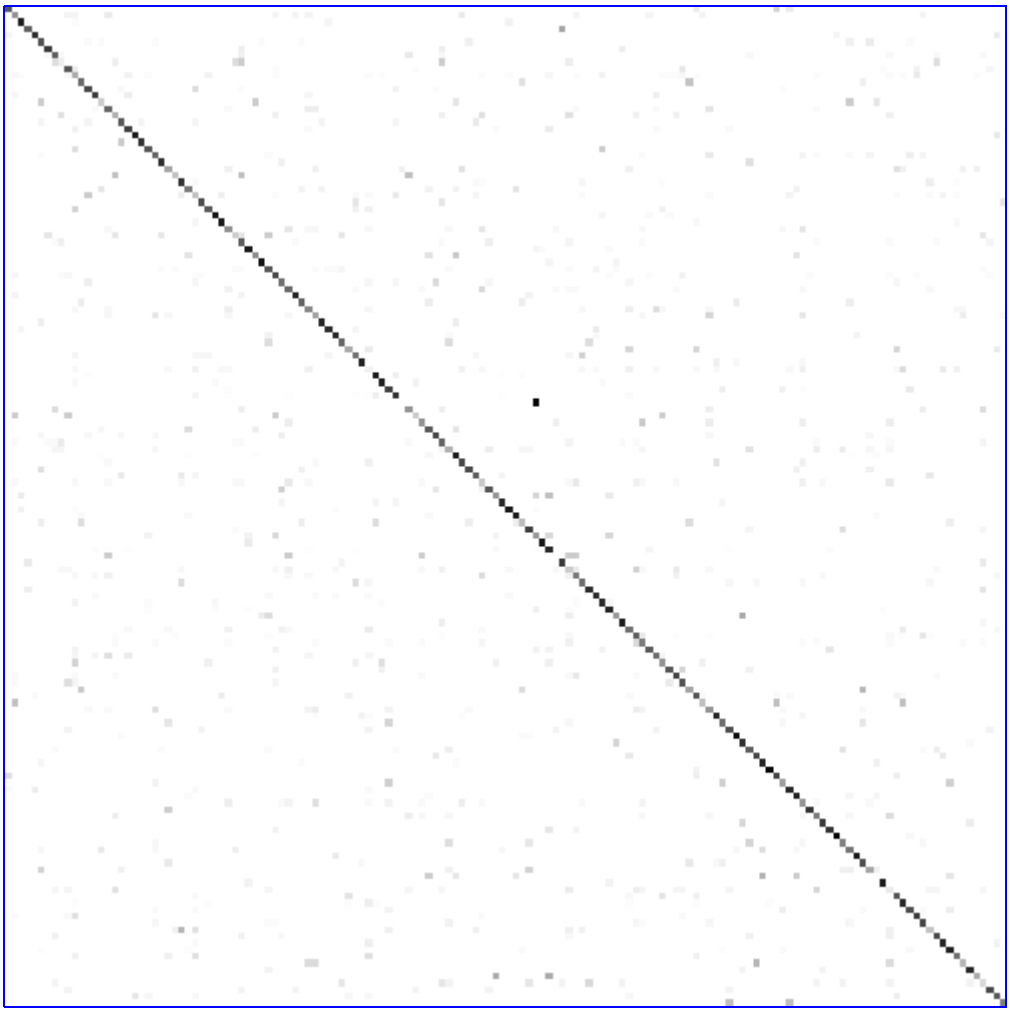


Figure 7.3.7 - Discrete Probability Density Function for ASL Model

Figure 7.3.7 shows the ASL PDF, which again has a heavy diagonal dominance. This dominance is when $\mathbf{argmax}_i\left(P\left(C_i^{t+1}\middle|C_j^t\right)\right)$ and $i=j$ i.e. the highest probability is that the PDM will usually stay within the present cluster. The assumption can therefore be made that within any local patch the model can iterate to a local solution. This confirms the assumption used when calculating the optimum model shape. This assumption also provides two benefits:

1. The iteration to convergence of any global optimisation technique can be enhanced by allowing each hypothesis to iterate to a better solution within the present cluster.
2. A smaller population is required, as only global differences in hypotheses need to be supported.

This is a common procedure in speeding up the convergence on solutions for many optimisation techniques such as in neural networks or clustering [Boyle 95]. By combining a gradient descent method with a global optimisation approach the speed to convergence is increased and the problem of oscillating down narrow energy wells to local minima reduced.

From the 'learnt' probability density function, a sample population can be generated at each iteration of the model. Given a good initialisation of the model (see section 3.3.2) and the associated cluster $C^{t=0}$, which encompasses that shape, the procedure is summarised thus:

Algorithm 7-1 - Simple CSSPDM Condensation

- From the PDF $P\left(C_i^t \middle| C_j^{t-1}\right)$, extract the probability vector $P\left(C_i^{t=1}\right)$, which is the probability distribution of the first iteration, given $C_j^{t-1} = C^{t=0}$.
- Generate a randomly sampled distribution of *k* hypothoses $\mathbf{x}_\rho [\rho = 1,\ldots,k]$, where $\mathbf{x}_\rho$ is the mean shape of cluster C_i and $P(C_i) = P\left(C_i^{t=1}\right)$
- While still tracking,
  - Fit the *k* hypothoses to the image frame using the least squares gradient descent algorithm (section 3.3) and iterate, applying CSSPDM constraints and assess fitness using error metric (section 7.3.2)
  - Sort hypothoses into descending order according to error
  - Take lowest error solution and locate closest cluster *c*
  - From the PDF $P\left(C_i^t \middle| C_j^{t-1}\right)$, extract the vector $P\left(C_i^t\right)$, which is the probability distribution of the next iteration, where $C_j^{t-1} = c$

- Generate a new randomly sampled distribution of k hypothoses $\mathbf{x}_\rho[\rho = 1,\ldots,k]$ where $\mathbf{x}_\rho$ is the mean shape of cluster C_i and $P(C_i) = P(C_i^t)$

By repeating this procedure for each frame, iteration allows the model to converge in the least square sense upon local solutions. However, due to the generation of a new population of hypotheses gained from the *a priori* information about movement contained within the PDF, the models are permitted to 'jump' within shape space at each new frame. This allows multiple hypotheses to be supported simultaneously, where the current lowest cost hypothosis is deemed to be the correct one. Figure 7.3.8 demonstrates the error rates produced by this simplified form of the condensation algorithm (Algorithm 7-1). Experiments were performed to assess the result of various parameterizations of the algorithm, where

n is the length of the normal search on either side of the contour
I is the number of least squares iterations used for each hypothosis
k is the size of the population size or the number of hypothosis used

Varying these parameters produces dramatic variations in the resulting error rates produced and the overall performance of tracking. Many of the higher error parameterizations fail to track the image sequence completely producing a zero success rate and hence consistently high error rates. With *n=40* (as with least squares iterative refinement) high failure rates are produced, as do small populations and low numbers of iterations. It is important to note that a population size of one (*k=1*) is effectively least squares iterative refinement due to the diagonal dominance of the PDF.

The best results were achieved using a normal length of 20 pixels, a population size of 10 multiple hypotheses and between 5 and 10 iterations per hypothesis (i.e. *n=20, k=10, I=5/10).* These traces are shown in Figure 7.3.9 along with the results of both the optimum trajectory and the iterative refinement approach for comparison. The trend lines give a good indication of the overall performance of the various approaches.

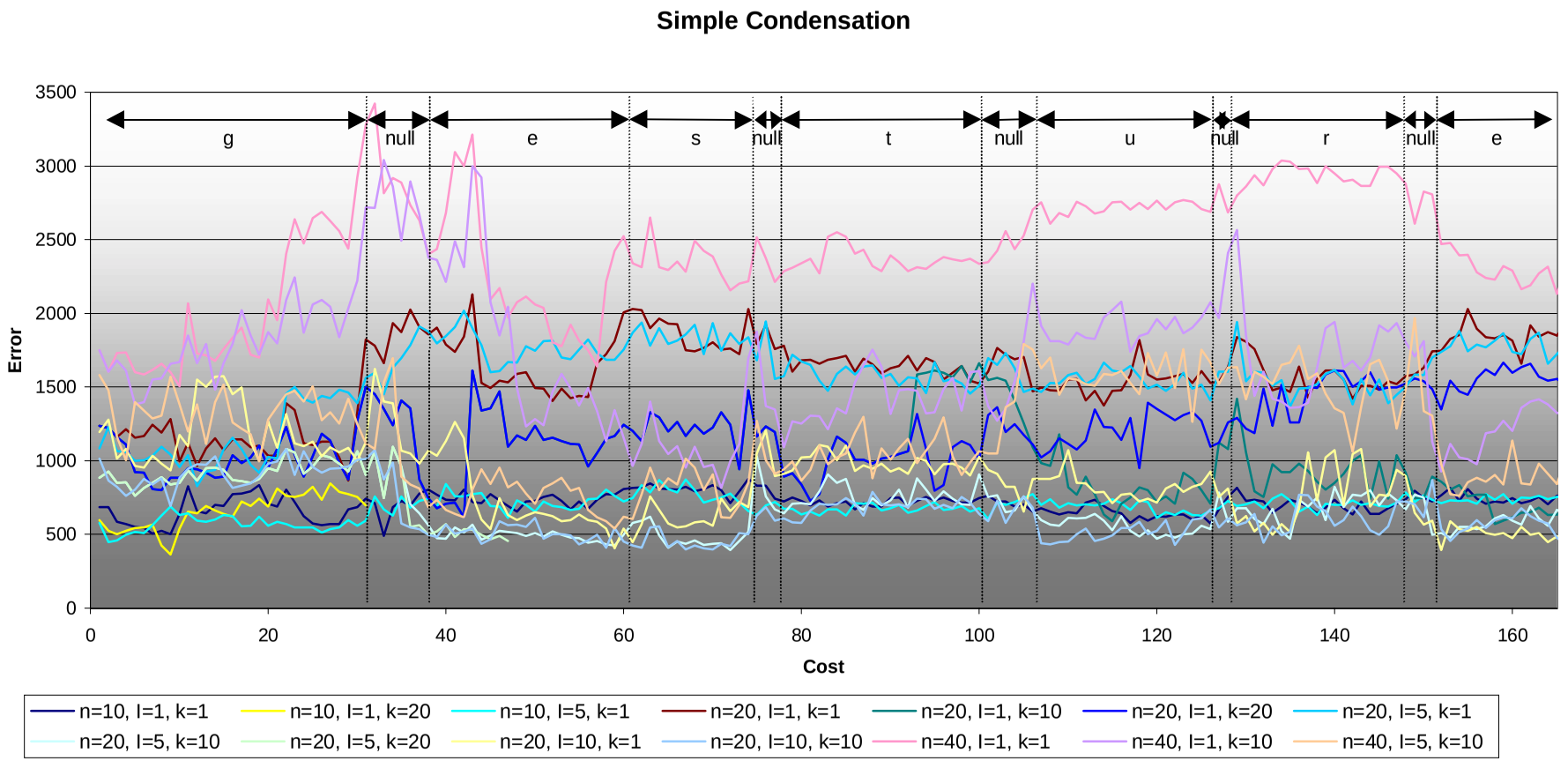


Figure 7.3.8 - Graph showing the Error rates Achieved by Varying the Parameters of the Simplified Condensation Algorithm

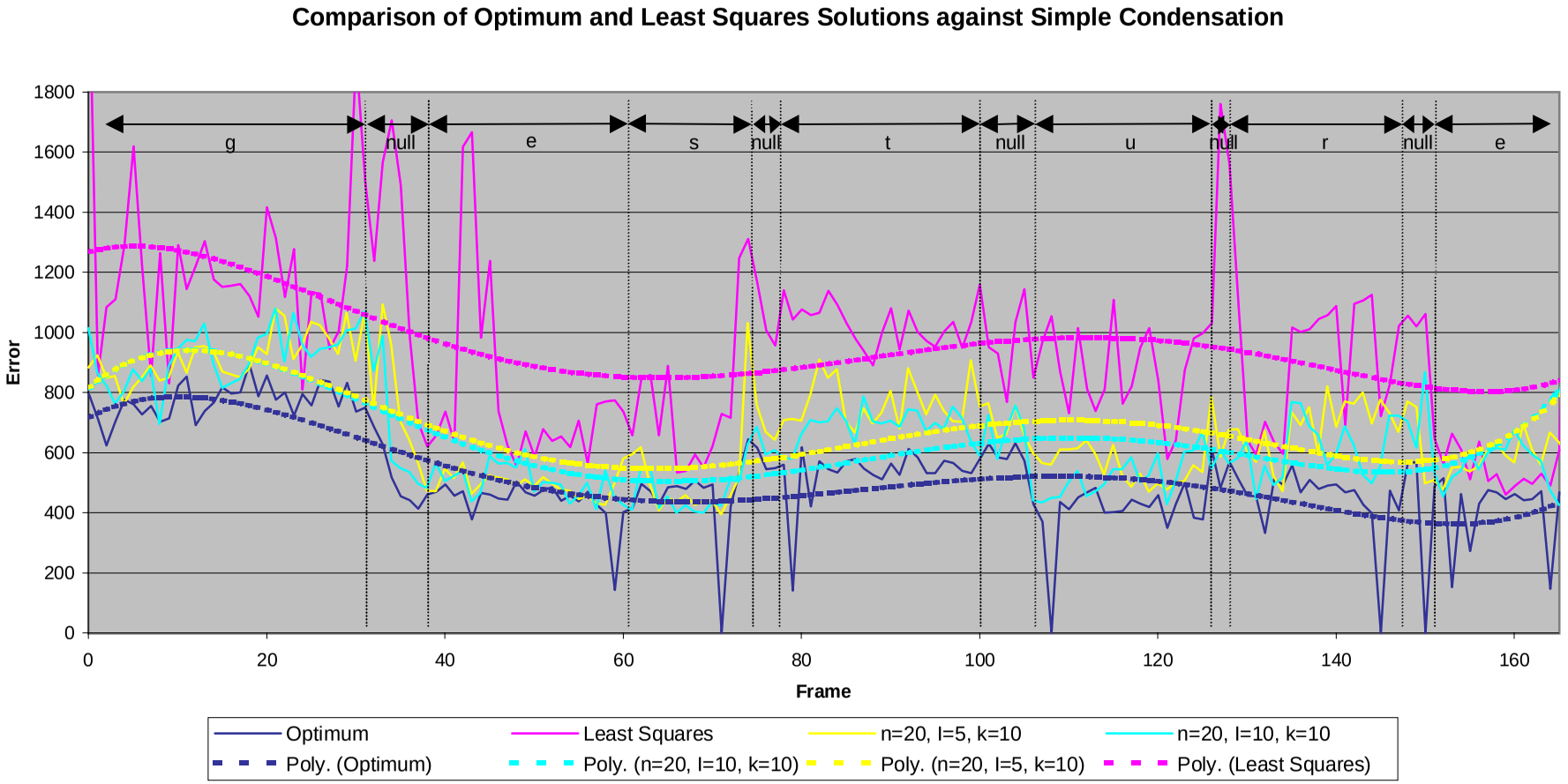


Figure 7.3.9 - Graph Comparing Simple Condensation against Previous Techniques

Figure 7.3.9 shows that both the simple condensation approaches produce significantly better results than the iterative refinement least squares tracking, but not as low as the optimum which would be expected. Increasing the number of iterations performed on each frame from 5 to 10 provides a slight increase in performance but not significant enough to warrant the additional computational overhead.

However, with such a low population size (p=10) and only five iterations required per frame (i=5) a total of (p*i), 50 models are fitted to the image at each frame. This provides a significant computational saving upon standard condensation where typically much larger populations (in the order of hundreds are required) to accurately track objects.

However, this approach, unlike condensation, does not recover well from failures. As the new population is solely based upon the current best-fit cluster the approach is highly sensitive to both an accurate PDF representation of the expected movement and the assumption that the best-fit cluster is actually affixed upon the object. To help overcome this drawback two factors must be addressed.

1. Less emphasis must be placed upon the current best-fit hypothesis being the optimum (and hence correct) solution, thus providing more robustness to failure.
2. The PDF must be an accurate and thorough representation of the expected object movement and hence the training set from which it is constructed must be general in both shape and movement. This is more difficult and will be addressed in the section 7.4.1.

Point 1 can be addressed by creating a new population of hypotheses, not from the current best fit model, but from the weighted sum of the best *n* hypotheses as described thus:

Algorithm 7-2 - Weighted Condensation

- From the PDF $P\left(C_i^t \middle| C_j^{t-1}\right)$, extract the probability vector $P\left(C_i^{t=1}\right)$, which is the probability distribution of the first iteration, given $C_j^{t-1} = C^{t=0}$.
- Generate a randomly sampled distribution of *k* hypothoses $\mathbf{x}_\rho [\rho = 1,\ldots,k]$, where $\mathbf{x}_\rho$ is the mean shape of cluster C_i and $P(C_i) = P\left(C_i^{t=1}\right)$
- While still tracking,
  - Fit *k* hypotheses, applying CSSPDM constraints and assess fitness using error metric
  - Sort hypotheses into descending order according to error
  - Iteratively refine first *n* hypotheses and resort
  - Apply the CSSPDM constraints and determine the *n* clusters C_η^{t-1}, where $\eta = 1,\ldots,n$ which produce the lowest error
  - From the PDF $P\left(C_i^t \middle| C_j^{t-1}\right)$, extract the vector $P\left(C_i^t\right)_\eta$ using the *n* extracted clusters. Take the weighted sum using a Gaussian weighting distribution to form a new distribution $P'\left(C_i^t\right)$, where

$$P'(C_i^t) = \sum_{\eta=1}^{n} \omega_\eta P(C_i^t)_\eta \text{ and } \omega_\eta = \exp\left[\frac{-9(1-\eta)^2}{2n^2}\right]$$

- Normalise probability distribution $P'\left(C_i^t\right)$.
- Generate a new random population of *k* hypotheses from the distribution $P'\left(C_i^t\right)$.

The results of applying this *weighted* approach to condensation are shown in Figure 7.3.10. This graph shows that, by using the best 5 models to generate the new population, lower error rates are achieved. Using the best 6 models produces less clear benefits but does provide increased ability to recover from failure.

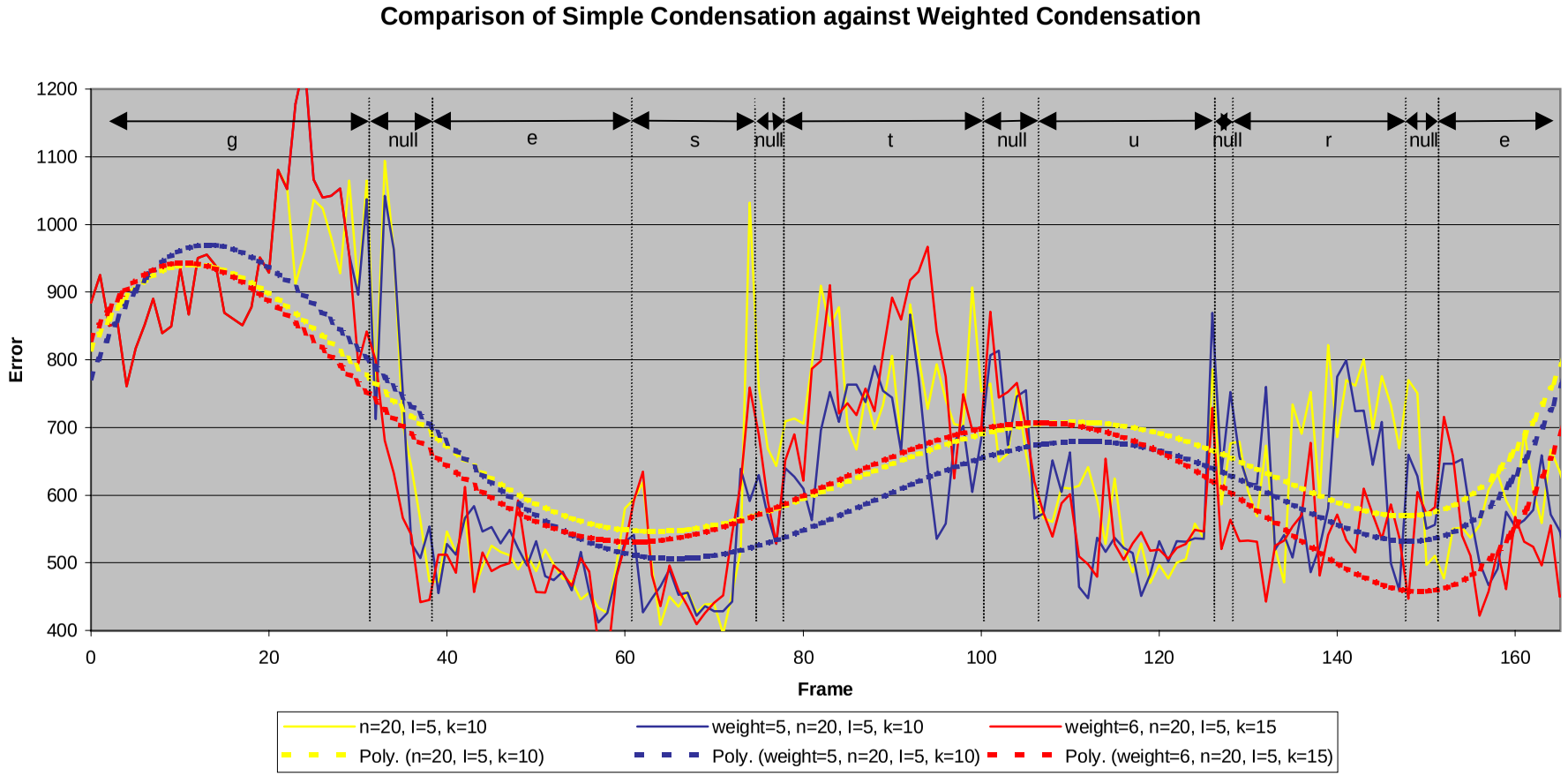


Figure 7.3.10 - Graph Comparing Simple Condensation against Weighted Condensation

7.3.4 Conclusion

This section has demonstrated that the nature of shape space need not be continuous. Under these circumstances it has been shown that the least squares, iterative refinement approach to PDM tracking fails. It has also been shown how the Markovian assumption can be applied to the CCSPDM to provide a fast tracking paradigm, which is less computationally expensive than standard condensation, while allowing multiple hypotheses to be supported.

7.4 Extending Temporal Dynamics to Classification

7.4.1 Introduction

It has been shown how, with the addition of a first order Markov chain to the CSSPDM, a hybrid approach to condensation can be used to provide robust tracking where either:

- The non-linearity of the PDM along with the discrete representation of the non-linear approximation leads to a discontinuous shape space.
- Rapid movement of the object produces large changes in the model parameters.

This Markovian model of dynamics can be used to explicitly constrain the movement of the model within shape space, or implicitly, using the hybrid condensation approach. However, the use of temporal constraints relies upon one major assumption, as mentioned earlier:

> ***The training set from which the model is built contains a thorough representation of all-possible deformation and movement.***

For simple models this is often true. However, for ASL it is not, and it is important to ask the question,

'What exactly is the temporal model representing?'

The ASL PDF represents two aspects of motion,

1. The non-linear representation of shape space, how the individual clusters relate and how the model moves throughout the space to form letters.
2. It also contains information about the English language and how letters relate to form words and sentences.

As the PDF encodes both of these attributes it must be constructed from a training set which has a good representation of how the model deforms and be representative of the English language. This is however infeasible.

If the ASL image sequence used previously is considered, it took 165 frames to record the 7 letter word 'gesture'. Konheim reported a statistical study where the 1-state transition probabilities of the English Language were determined using 67,320 transitions between two successive letters [Konheim 82]. As the 165 frames previously used produced an average of 20 frames per letter, this would constitute a training set in excess of 1.3 million frames not including transitional shapes between letters. As each frame produces a training shape this results in a training set which is of infeasible size. At 12.5 frames per second it would require almost 30 hours of continuous video capture. Of course smaller numbers of both transitions and frame sampling could be used but would result in a less reliable PDF.

The current ASL PDF (see Figure 7.3.7) contains valuable information about how the model moves within shape space, but due to the deficiency in training it does not contain sufficient information to accurately model the transitions between the letters of the English language. Fortunately, it is relatively simple to gain a transition matrix for the English language as it can be constructed in a similar manner to previously described PDF's by analyzing large samples of electronic text and calculating the 1-state transitions. What is required is a method of combining this knowledge of English into the ASL PDF, producing a more generic and accurate model for tracking and classification.

7.4.2 The Temporal Model

The ASL PDF $P\left(C_i^t \middle| C_j^{t-1}\right)$, constructed from the training set, provides the probability that the model will move to cluster C_i given it was at cluster C_j at the last time step. This is illustrated by Figure 7.4.1, and provides the necessary information of how the model moves within shape space. However, as discussed, this information is incomplete and does not correctly contain the transitional information about the letters and how they relate to form words.

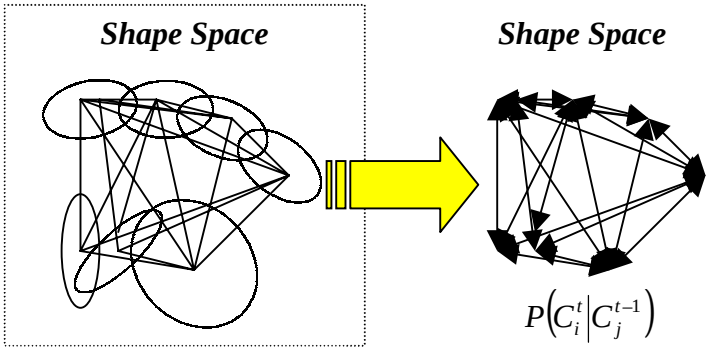


Figure 7.4.1 - Temporal Constraints upon Shape Space for the ASL Model

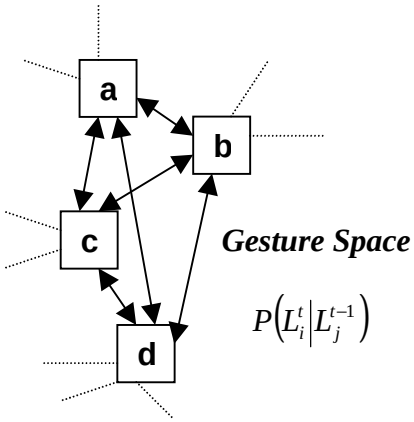


Figure 7.4.2 - 1st Order Markov Chain in Gesture Space

Similarly a 1st order Markov Chain can be constructed for the English language which provides a new PDF $P\left(L_i^t \middle| L_j^{t-1}\right)$ (see Figure 7.4.2). Figure 7.4.3 shows the PDF gained from this Markov Chain as taken from Konheim and shows the 1-

state transitions calculated from a sample text of over 67 thousand letters [Konheim 82].

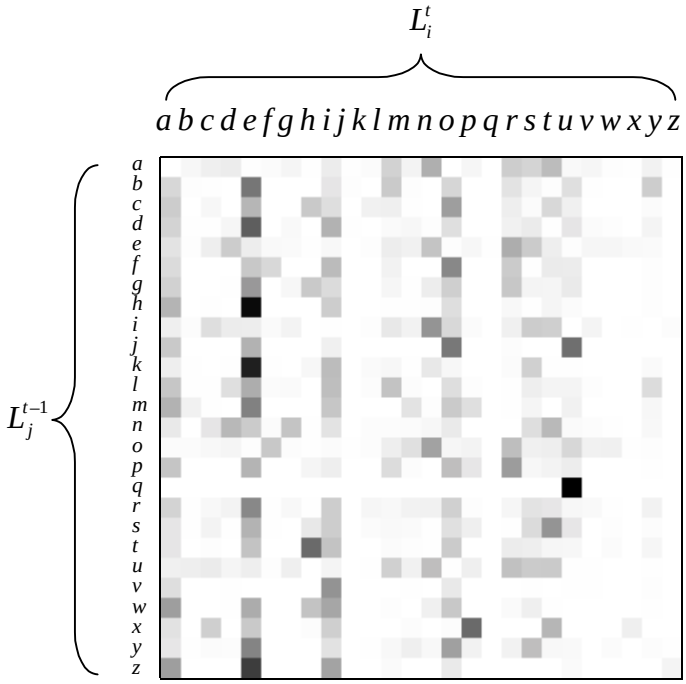


Figure 7.4.3 - Discrete Probability Density Function for the English Language

Figure 7.4.3 does not demonstrate a diagonal dominance, unlike previous PDF's. This is because the English language has few occurrences of repetitive letters in words whereas previous PDFs resulted from operations involving a high degree of repetition. The main trend that can be seen are the vertical stripes that occur for many of the letters. This shows letters which have a high occurrence and are proceeded by almost any other letter in the alphabet. The highest probabilities occur for the letter 'e' confirming that 'e' is the most commonly used letter in the English language. Another observation is the single transition from the row 'q' to the column 'u' as 'q' is always followed by a 'u' in standard English.

In order to incorporate this additional information learnt from sample text, a new ASL PDF must be constructed $P'\left(C_i^t \middle| C_j^{t-1}\right)$. To do this a mapping must be achieved which allows shape space to relate to gesture space.

7.4.3 Extending to a Hidden Markov Model

It has already been shown how a mapping can be achieved between the gesture space and shape space for use in classification (see section 6.5). Here the conditional probability $P\left(L_i^t \middle| C_j^t\right)$ provides a probability of the occurrence of a letter *L* given the model is in cluster *C* in shape space at any time.

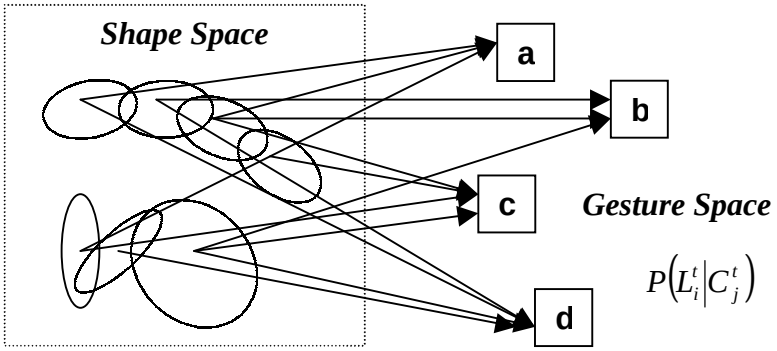


Figure 7.4.4 - Conditional Probabilities Connecting Cluster Exemplars in Shape Space to Specific Letters in Gesture Space

This conditional probability provides a mechanism to relate the shape space to the gesture space where the constraints of the English language (as learnt) can be applied. However, for this to be of use, a method that allows this information to be mapped back into the shape space must be provided. This can be done using the common form of Bayes theorum,

$$P\left(A \middle| B\right) = \frac{P(A)P\left(B \middle| A\right)}{P(B)} or \frac{P(A)P\left(B \middle| A\right)}{\sum P(A)P\left(B \middle| A\right)}$$

Therefore, placing this in the context of the ASL CSSPDM

$$P\left(C_i^t \middle| L_j^t\right) = \frac{P\left(C_i^t\right)P\left(L_j^t \middle| C_i^t\right)}{P\left(L_j^t\right)}$$

However, where $P\left(C_i^t \middle| L_j^t\right)$ and $P\left(C_i^t\right)$ can both be gained from the training set, $P\left(L_j^t\right)$ (the probability of the occurrence of a letter) can only be gained from analyzing English text. As it is known that the training set does not fully represent the English Language this equation would lead to biasing of the final

conditional probabilities. Instead, a variation of Bayes Theorem can be used, where

$$P\left(C_i^t \middle| L_j^t\right) = \frac{P\left(C_i^t\right)P\left(L_j^t \middle| C_i^t\right)}{\sum P\left(C_i^t\right)P\left(L_j^t \middle| C_i^t\right)}$$

Using this form, $\sum P\left(C_i^t\right)P\left(L_j^t \middle| C_i^t\right) \equiv P\left(L_j^t\right)$ but all probabilities are gained from the training set, and hence no bias occurs from mixing unrelated probabilities. This is possible as, although the training set does not contain a thorough representation of English, it does provide an accurate representation of the mapping between the two spaces.

7.4.4 Augmenting the Hidden Markov Model to Increase Constraints

All the necessary tools are now available which allow a new ASL PDF to be constructed which incorporates the 1-state transitions of the English Language.

- $P\left(L_i^t \middle| C_j^t\right)$, is the conditional probability that the model is representing a letter *L* at time *t*, given the CSSPDM is in cluster *C* and time *t*.
- $P\left(C_i^t\right)$, is the probability of the occurrence of cluster *C*.
- $P\left(C_i^t \middle| L_j^t\right) = \frac{P\left(C_i^t\right)P\left(L_j^t \middle| C_i^t\right)}{\sum P\left(C_i^t\right)P\left(L_j^t \middle| C_i^t\right)}$, is the conditional probability that the CSSPDM is in cluster *C* at time *t*, given the current letter that is being represented is *L*.
- $P\left(L_i^t \middle| L_j^{t-1}\right)$, is the 1-state transition that a letter L_i will occur given the previous letter was L_j.

A new ASL PDF can therefore be constructed which incorporates the 1-State transitions of English, by

1. Taking the current cluster of the model
2. Calculating the corresponding letter(s) associated with this cluster
3. Applying the 1-state transition matrix to extract the most likely next letter
4. Then locating the cluster(s) associated with this transition.

Where,

$$P'\left(C_i^t \middle| C_j^{t-1}\right) = P\left(L_i^t \middle| C_j^t\right) P\left(L_i^t \middle| L_j^{t-1}\right) P\left(C_i^t \middle| L_j^t\right)$$

This produces a new ASL PDF which is shown in Figure 7.4.5.

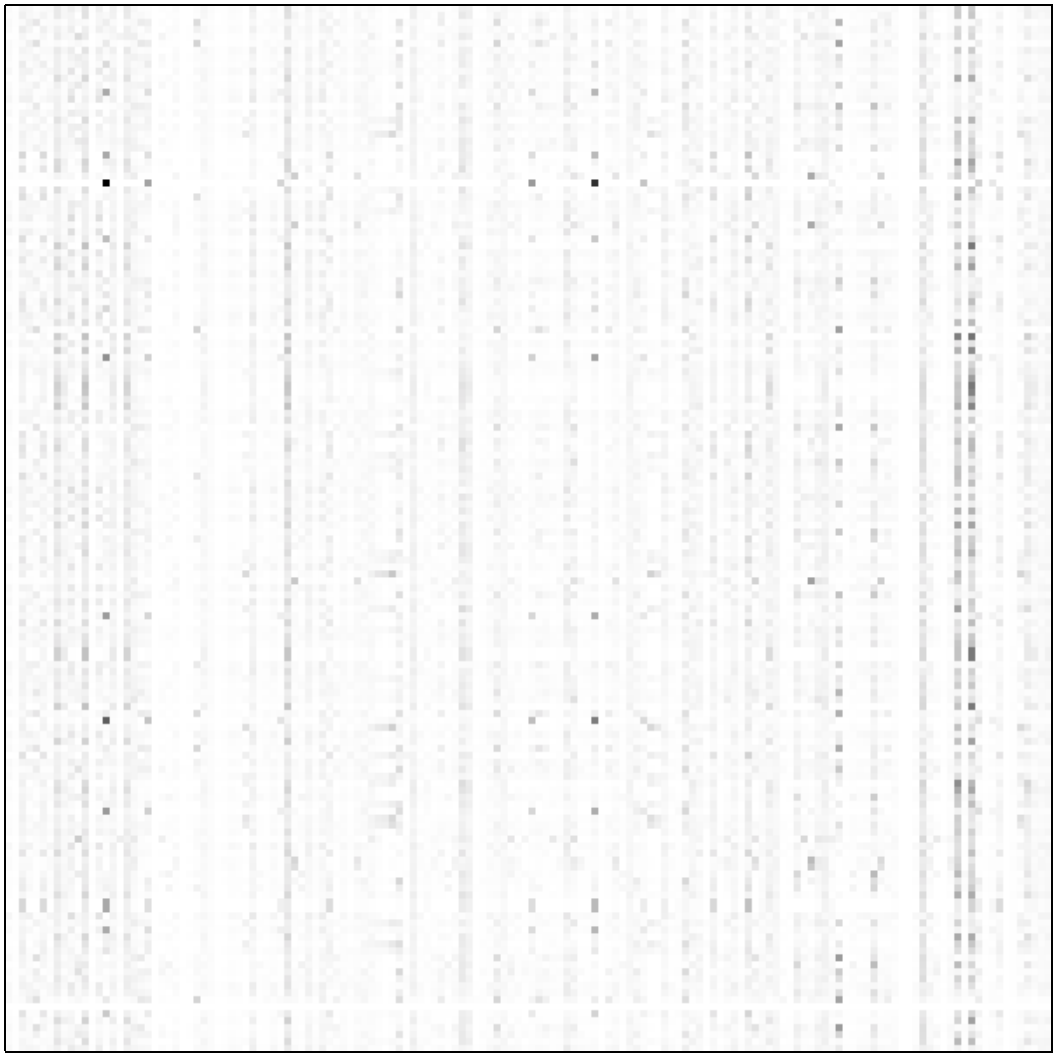


Figure 7.4.5 - Discrete Probability Density Function for derived ASL Model

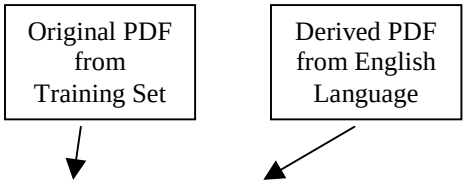
Figure 7.4.5 demonstrates the same characteristic vertical strips seen from the English Language PDF, which it has inherited, and as such differs from the original ASL PDF in two ways.

1. Each cluster exhibits far more transition to other clusters.
2. The diagonal dominance, which is important to tracking, is missing.

Diagonal dominance can be forced upon the PDF by imposing diagonal dominance on either $P\left(L_i^t \middle| L_j^{t-1}\right)$ or $P'\left(C_i^t \middle| C_j^{t-1}\right)$. However, this is haphazard and risks over-biasing the hypothesis generated at each frame. An alternative is to simply ensure that the population generated at each step always includes at least one hypothesis from the current cluster.

In order to explore the validity of these assumptions and assess the success of the derived PDF a new set of tests were performed upon the 'gesture' image sequence.

The PDF used for each test was the weighted sum of the original PDF gained from the training set and the derived PDF from English, where



$$P''\left(C_i^t \middle| C_j^{t-1}\right) = (1-\alpha)P\left(C_i^t \middle| C_j^{t-1}\right) + \alpha P'\left(C_i^t \middle| C_j^{t-1}\right), \text{ for } 0 \le \alpha \le 1$$

and hence

$$P''\left(C_i^t \middle| C_j^{t-1}\right) = (1-\alpha)P\left(C_i^t \middle| C_j^{t-1}\right) + \alpha P\left(L_i^t \middle| C_j^t\right)P\left(L_i^t \middle| L_j^{t-1}\right)P\left(C_i^t \middle| L_j^t\right)$$

Using this method, the performance of both approaches can be assessed. Figure 7.4.6 shows the results of varying α. When $\alpha = 0$ the PDF is that gained from the training set; but as α increases, the resultant error rate decreases. When $\alpha = 0.6$ the resulting error rate is only slightly higher than that produced by the optimum path shown in Figure 7.3.4. However, as α approaches 1 an increase in error rate results. This is attributable to the absence of diagonal dominance for the derived PDF, and hence lack of support for hypotheses that remain static within shape space. However, even in light of this fact, the overall error is still lower than that gained form the original ASL PDF.

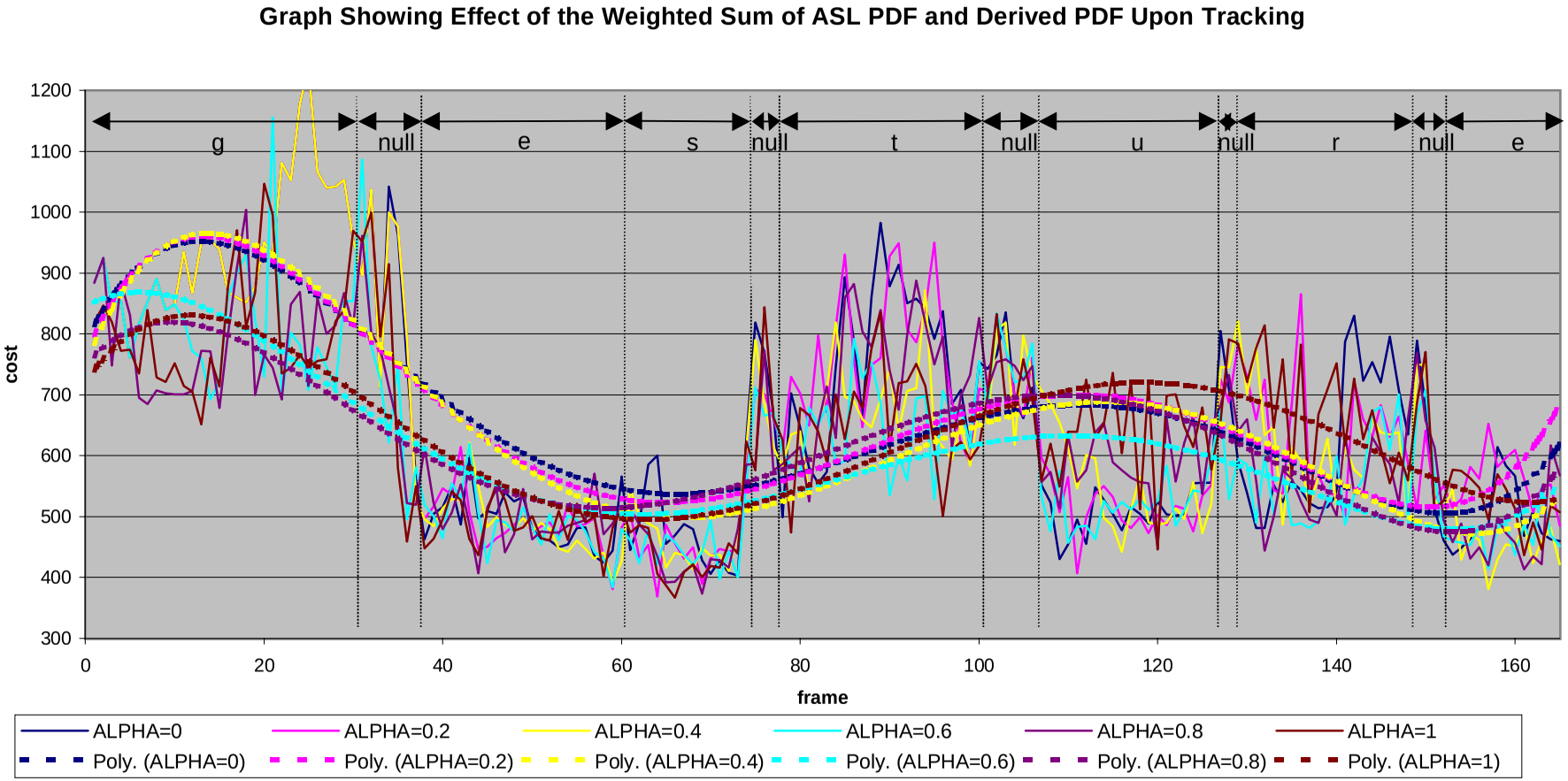


Figure 7.4.6 - Graph Comparing Simple Condensation Against Weighted Condensation

7.5 Conclusions

This chapter has looked at augmenting statistical models with temporal dynamics gained through the probabilistic analysis of the training set and how this relates to movement within shape space. It has been shown how the discrete segregation of shape space used in the CSSPDM directly lends itself to a Markov chain approach to modeling temporal dynamics. This additional analysis has been used to reproduce motion indicative of the training sets in the form of key frame animations and how the motion of the CSSPDM can be further constrained during tracking. It has been shown that the nature of shape space is often complex and discontinuous and how, using these additional learnt temporal constraints, tracking can be improved by supporting a population of multiple hypotheses. Lastly a method of combining additional constraints into the model was presented which provides more robust tracking and classification, while reducing the necessity for large training sets.