# A Probabilistic Framework for Matching Temporal Trajectories: CONDENSATION-Based Recognition of Gestures and Expressions

Michael J. Black[1] and Allan D. Jepson[2]

[1] Xerox Palo Alto Research Center,
3333 Coyote Hill Road, Palo Alto, CA 94304 USA.
black@parc.xerox.com

[2] Department of Computer Science, University of Toronto,
Toronto, Ontario M5S 1A4 Canada.
jepson@vis.toronto.edu

**Abstract.** The recognition of human gestures and facial expressions in image sequences is an important and challenging problem that enables a host of human-computer interaction applications. This paper describes a framework for incremental recognition of human motion that extends the "CONDENSATION" algorithm proposed by Isard and Blake (ECCV'96). Human motions are modeled as *temporal trajectories* of some estimated parameters over time. The CONDENSATION algorithm uses random sampling techniques to incrementally match the trajectory models to the multi-variate input data. The recognition framework is demonstrated with two examples. The first example involves an augmented office white-board with which a user can make simple hand gestures to grab regions of the board, print them, save them, etc. The second example illustrates the recognition of human facial expressions using the estimated parameters of a learned model of mouth motion.

## 1 Introduction

Motion is intimately tied with our behavior; we move when we communicate through facial expressions and gestures. The estimation and explanation of this sort of human motion is a challenging problem with diverse applications in human-computer interaction, medicine, robotics, animation, video databases, and surveillance to name a few. In this paper we focus on the problem of recognizing human motion using a probabilistic framework that is based on random sampling techniques. We represent human motions as *temporal trajectories* and we match these to estimated trajectories in an on-line fashion. This framework applies the CONDENSATION algorithm [8, 9] in a novel way to recognize complex human motions. We illustrate the method with examples of human gestures and human facial expressions.

Our focus here will be on recognition of gestures given some estimated representation of the human motion. For each pair of frames in a video sequence we compute a set of parameters that describe the motion. These might be simply horizontal and vertical velocity or the parameters of a more sophisticated parameterized model of optical flow [3]. Over time the estimated parameter vectors form *temporal trajectories* that characterize the gesture or expression. Given a number of examples of a gesture we construct a model of the temporal trajectory that encodes the mean trajectory and the expected variance along the trajectory. This model may be a single trajectory (discretely sampled or continuous) or a collection of such trajectories.

For a new image sequence, recognition is performed in an "on-line" fashion. As new parameter vectors are estimated, we incrementally match our trajectory models to the data. The input data is typically noisy and may deviate from the stored model in a variety of ways. When different subjects perform the same action, the recovered motion parameters will vary and the matching must account for this variability. Additionally, we do not know where activities begin or end so the algorithm must segment the input data automatically during recognition.

We have a variety of gestures or expressions that we wish to recognize and at a given instant in time the interpretation of the input data may be ambiguous. To accommodate this fact, our probabilistic framework maintains multiple hypotheses and automatically focuses more computational resources on the more likely hypotheses.

Our approach applies the CONDENSATION algorithm to the problem of recognizing temporal trajectories. The CONDENSATION (CONDitional dENSity propagATION) algorithm uses random sampling techniques to simply and elegantly search a multi-variate parameter space that is changing over time. The algorithm was proposed by Isard and Blake [8] for tracking objects in clutter and has recently been extended [9] to simple gesture-recognition tasks. Here we extend the method further to recognize more complex temporal activities. See also [11] for a related random sampling techinque used in dynamic probabilistic belief networks.

We define a "state" to be a set of parameters that align a trajectory model with the input data. A state includes parameters that control local stretching, scaling, and translation of the model with respect to the data. We define the probability of a state in terms of how well it matches the data. Then a set of states and their probabilities defines a discretely sampled probability distribution over the match parameters. This distribution evolves over time as the input data changes. The CONDENSATION algorithm uses stochastic dynamics and random sampling techniques to "track" this distribution as it evolves. We refer to the recognition framework as CONDENSATION-based Trajectory Recognition (CTR).

To illustrate the CTR framework we provide two examples. In the first we use gesture data gathered by a computer vision system that is observing an office whiteboard. We describe a vocabulary of gestures that a user can perform at the whiteboard to extend its functionality. The second example illustrates the recognition of facial motions from optical flow parameters.

The following section reviews related work on the CONDENSATION algorithm, and gesture recognition. Section 3 presents the algorithm in detail. Sections 4 and 5 present our two examples.

## 2 Previous Work

The recognition of temporal trajectories has received extensive study in the contexts of gesture, speech, and handwriting recognition. We do not attempt an exhaustive review here but rather highlight the relationships between the proposed framework and the most relevant related approaches. The two most common methods for recognizing temporal trajectories are Hidden Markov Models (HMM's) [14] and Dynamic Time Warping (DTW) [16]. There have been many applications of these basic techniques to the problem of gesture recognition in computer vision (e.g. [6, 18, 20]).

The proposed method can be seen as a generalization of HMM's in that it allows a discrete set of states with probabilistic transitions between states. With CTR, however, the recognition of the individual states involves the probabilistic matching of an entire temporal trajectory model that represents a portion of the gesture. In this way the CTR method uses the entire curve in a way similar to DTW but within a unified probabilistic framework. The temporal trajectories may be discrete or continuous and the method allows parameterized deformations of the trajectories (see also [21]). Computational resources are automatically applied to more fully explore the most probable matches.

The "EigenCurve" representation [22] has been proposed as an alternative to HMM's and DTW. The approach matches an input trajectory to an basis-set representation of stored curves while allowing various global deformations. The CTR framework could be extended to represent temporal trajectories using this "eigen" basis representation. The advantages over [22] would include on-line recognition, automatic segmentation, and local curve deformations.

The most significant advantage of the CTR approach is that it allows the formulation of recognition and motion tracking in the same probabilistic framework. Isard and Blake [8] initially developed the CONDENSATION algorithm to deal with the difficult problem of tracking an object in a cluttered environment. In this case the algorithm is able to maintain a probability distribution over multiple tracking hypotheses which provides robustness while achieving near real-time performance.

In [9] they also show how this tracking ability can be combined with simple dynamical models of gestures to simultaneously perform tracking and recognition. They define a "mixed-state" representation that adds a discrete model parameter to the definition of a state. This allows the tracker to use multiple motion models corresponding to different gestures. The recognition in [9] is limited to fairly simple gestures (e.g. "scribbling" motion versus "smooth" motion).

In this paper we focus only the recognition part of the problem and by incorporating explicit, learned, temporal trajectories we extend the CONDENSATION method to more complex gestures. Our current research is focused on combin-

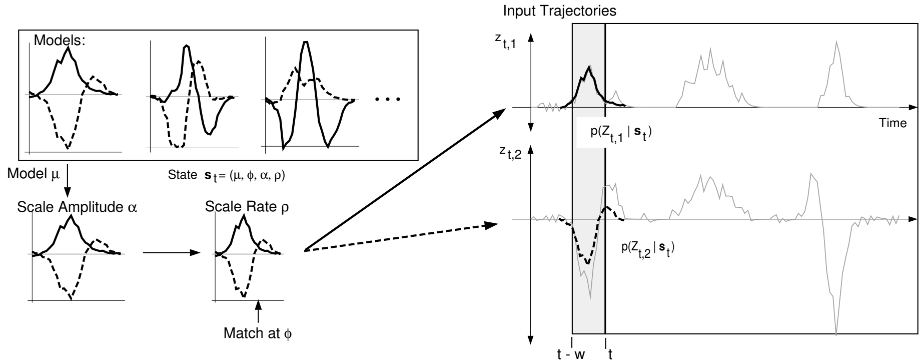


**Fig. 1.** Our goal is to incrementally match $M$, multiple-parameter, trajectory models to input data.

ing tracking and recognition thus allowing the temporal models to constrain the tracking problem (cf. [23]).

## 3 Condensation Algorithm

Our goal is to take a set of $M$ model trajectories $\{\mathbf{m}^{(\mu)}, \mu = 1, \ldots, M\}$ and match them against an $N$-dimensional input trajectory, given by $\mathbf{z}_t = (z_{t,1}, \ldots, z_{t,N})$ at time $t$ (see Figure 1). The models are taken to be discretely sampled curves (though they may be continuous as well) with a phase parameter $\phi \in [0, \phi_{\max}]$ representing the current position in the model. The model values at position $\phi$ are a vector of $N$ values $\mathbf{m}_\phi^{(\mu)} = (m_{\phi,1}^{(\mu)}, \ldots, m_{\phi,N}^{(\mu)})$ where the stored discrete curve is linearly interpolated at phase $\phi$.

The parameters we need to estimate to match a model to the data are:

$\mu$: an integer indicating the model type that is being matched,
$\phi$: position (or phase) within the model that aligns it with the data at time $t$,
$\alpha$: an amplitude parameter used to scale the model vertically to match the data, and
$\rho$: a rate parameter that is used to scale the model in the time dimension.

We define a state at time $t$ to be a vector of parameters $\mathbf{s}_t = (\mu, \phi, \alpha, \rho)$.

We would like the find the state $\mathbf{s}_t$ that is most likely to have given rise to the observed data $Z_t = (\mathbf{z}_t, \mathbf{z}_{t-1}, \ldots)$. Let $Z_{t,i} = (z_{t,i}, z_{(t-1),i}, z_{(t-2),i}, \ldots)$ be the vector of observations for the $i^{th}$ coefficient over time. We define the likelihood of an observation $\mathbf{z}_t$ given the state $\mathbf{s}_t$ as

$$p(\mathbf{z}_t | \mathbf{s}_t) = \Pi_{i=1}^N p(Z_{t,i} | \mathbf{s}_t), \tag{1}$$

where

$$p(Z_{t,i} | \mathbf{s}_t) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp \frac{-\sum_{j=0}^{w-1} (z_{(t-j),i} - \alpha m_{(\phi-\rho j),i}^{(\mu)})^2}{2\sigma_i^2 (w-1)}$$

and where $w$ is the size of a temporal window backwards in time over which we want the curves to match. The $\sigma_i$ are estimates of the standard deviation for curve $i$. Also, $\alpha m^{(\mu)}_{(\phi-\rho j),i}$ is simply the value of the $i^{th}$ coefficient in the model $\mu$ interpolated at time $\phi - \rho j$ and scaled by $\alpha$.

Given a definition for $p(\mathbf{z}_t|\mathbf{s}_t)$, we can construct a discrete representation of the entire probability distribution over the possible states. Initially, we sample uniformly for the state parameters

$$\begin{aligned}
&\mu \in [0, \mu_{\max}] \\
&\phi = \frac{1-\sqrt{y}}{\sqrt{y}}, \text{ where } y \in [0,1] \\
&\alpha \in [\alpha_{\min}, \alpha_{\max}] \\
&\rho \in [\rho_{\min}, \rho_{\max}].
\end{aligned}$$

Note that the prediction for the initial phase $\phi$ is biased towards small values. We then compute the probability of the state $p(\mathbf{z}_t|\mathbf{s}_t)$. We can do this for $S$ samples where we take $S$ on the order of 1000. This gives us a set $\{\mathbf{s}_t^{(n)}, n = 1, \ldots, S\}$ of samples. We normalize these probabilities so that they sum to one, producing weights $\pi_t^{(n)}$

$$\pi_t^{(n)} = \frac{p(\mathbf{z}_t|\mathbf{s}_t^{(n)})}{\sum_{i=1}^{S} p(\mathbf{z}_t|\mathbf{s}_t^{(i)})} \tag{2}$$

The CONDENSATION algorithm [8, 9] uses this information (the sample states and their weights) to predict the entire probability distribution at the next time instant. Unlike traditional tracking methods (e.g. Kalman filtering) this approach can deal well with ambiguous data since multiple matches are propagated in time. The algorithm has three stages (selection, prediction, updating). Below we outline how to construct a new probability distribution with $S$ samples at time $t$ given the the distribution at time $t-1$.

**1. Selection:** First we choose a state from time $t-1$ *according to the probability distribution* at time $t-1$. That is, we use the current probability distribution over states to choose likely states to predict into the future.

This is done by constructing a cumulative probability distribution using the $\pi_{t-1}^{(n)}$ as illustrated in Figure 2. Let the cumulative probabilities be

$$\begin{aligned}
c_{t-1}^{(0)} &= 0 \\
c_{t-1}^{(n)} &= c_{t-1}^{(n-1)} + \pi_{t-1}^{(n)}.
\end{aligned}$$

We sample this distribution by uniformly choosing a value, $r$, between zero and one. We then find the smallest $c_{t-1}^{(n)}$ such that $c_{t-1}^{(n)} > r$. The state $\mathbf{s}_{t-1}^{(n)}$ is then selected for propagation to the next time instant.

With this sampling method, states that are most probable will be selected with the highest frequency; this has been called the "survival of the fittest"

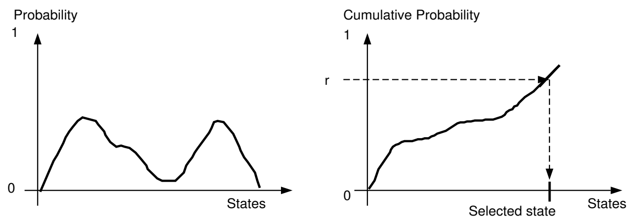


**Fig. 2.** We sample from the distribution over the states by constructing a cumulative probability distribution and sampling it uniformly. Choosing $r$ from a uniform distribution over [0, 1] gives us a corresponding state.

algorithm [11]. To avoid getting trapped in local maxima and to deal with sudden changes in the input data, we randomly choose some fraction of the states to be replaced by random initial guesses (initialized as described above). We typically take these random guesses to be $5 - 10\%$ of the samples.

**2. Prediction:** Given a state $\mathbf{s}_{t-1}^{(n)}$ we predict the parameters of the new state $\mathbf{s}_t^{(n)}$ at time $t$ to be

$$\begin{aligned}
\mu_t &= \mu_{t-1} \\
\phi_t &= \phi_{t-1} + \rho_{t-1} + \mathcal{N}(\sigma_\phi) \\
\alpha_t &= \alpha_{t-1} + \mathcal{N}(\sigma_\alpha) \\
\rho_t &= \rho_{t-1} + \mathcal{N}(\sigma_\rho)
\end{aligned}$$

where $\mathcal{N}()$ is a normal distribution and the $\sigma_*$ represent uncertainty in the prediction. Note that prediction can be viewed as sampling from the probability distribution $p(\mathbf{s}_t^{(n)}|\mathbf{s}_{t-1}^{(n)})$ [9]. For the time being, the model type $\mu$ does not change over time; we will extend the method below to allow transition probabilities to take $\mu$ to a new state.

During prediction, if $\phi_t > \phi_{\max}$ then that model has been completed and the state is initialized as described above. For the other parameters, we draw samples from $\mathcal{N}$ until the predicted value of the parameter is within its allowed range.

It is interesting to note that the $\sigma_*$ are used as a tool for locally searching the parameter space. They can be thought of as "diffusion" parameters that blur the probability distribution as it is predicted in time. They provide a way of performing a local search about a state. They also allow local deformations of the trajectories within the moving window of size $w$.

**3. Updating:** Given a new state we evaluate the probability, $p(\mathbf{z}_t|\mathbf{s}_t^{(n)})$, that it generated the data at time $t$ using Equation (1). If the likelihood is zero (or below a threshold) then we return to Step 2 for a new prediction. We repeat this for a fixed number of tries. If no prediction from $\mathbf{s}_{t-1}^{(n)}$ has sufficient probability then we generate a random initial sample.

After $S$ new states have been generated, we compute the normalized weights, $\pi_t^{(n)}$, using Equation (2) and repeat the process for the next time instant.

Note that the CONDENSATION algorithm is a probabilistic hybrid of depth-first and breadth-first search. When no good match to the input data is found, the method resorts to uniform sampling (breadth-first). When the probability mass is centered around a set of parameters, more resources will automatically be spent to explore the neighborhood around these parameters (depth-first). Also, note that a "simulated annealing" method could be used in the search to start with high values of $\sigma_i$ in Equation (1) and lower them over time.

### 3.1 Finite State Extension

We have also extended the method to allow compound models that are very like Hidden Markov Models in which each state in the HMM is one of our trajectory models (see also [9]). Let $\mathcal{M} = \{\mu_1, \mu_2, \ldots, \mu_i\}$ be a "parent state" which consists of a set of model types $\mu_i$ along with transition probabilities $p(\mu_t|\mathcal{M}, \mu_{t-1}, \phi_{t-1})$ between states. In the prediction step above, $\mu_t$ is chosen by sampling from the transition probabilities. When initializing a new random state, we first chose the parent type $\mathcal{M}$ from a uniform distribution. The initial model type $\mu_1$ is defined by the parent. The remaining parameters are chosen as described above. Currently transition probabilities are set by hand; learning these probabilities is a topic of current research.

## 4 Gesture Models

To test the CONDENSATION-based trajectory recognition algorithm we consider the problem of recognizing a set of gestures in the context of an augmented whiteboard. A number of authors have looked at problem of scanning whiteboards at high resolution using mosaicing [19] and interacting with the board by making hand-drawn marks [17]. Here we look at the problem of recognizing dynamic gestures. Isard and Blake [9] used the CONDENSATION algorithm to recognize simple drawing gestures. Our extension to temporal trajectories allows more complex gestures to be recognized.

In our scenario, when the user wants to perform a command, they pick up a gesture "phicon" (or physical icon) [10] that has a distinctive color that makes it easy to locate and track. The motion of the phicon is tracked using a color histogram tracker [7] in real time. Tracking is performed at roughly 30Hz. Since the tracking rate varies slightly, we resample the phicon locations at fixed time instants using linear interpolation. The horizontal and vertical velocity of the phicon is used for gesture recognition.

We define the following set of simple gestures which are useful for such a purpose (see Figure 3):

- **Start**: The start gesture tells the system to "pay attention" and start recognizing gestures. This gesture is simply a waving motion similar to what a person might do to get a human's attention.

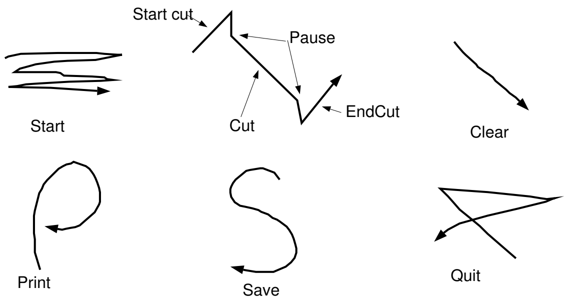


**Fig. 3.** Gestures.

- **Cut Region:** The cut gesture is used to indicate a region of the whiteboard to be scanned (possibly at higher resolution). This gesture consists of three primitive gestures with pauses in between of arbitrary duration. We refer to the complete cut gesture as a "parent" gesture that is made up of trajectory models.
  - **Cut-On**: The gesture begins with an upside-down "check mark." The end of this *cut-on* gesture marks the upper left corner of the scanning region.
  - **Cut**: The user then moves the phicon in a relatively straight line to the lower right corner of the region.
  - **Cut-Off**: To end the gesture and cut the image region the user makes a right-side-up "check mark".
- **Print**: To send a cut region to the printer, the user makes a gesture like the letter "P".
- **Save**: To save the region to a file, the user makes a gesture like the letter "S".
- **Clear**: A sharp diagonal motion "clears" the current stored whiteboard region. One can think of a cut region as being "stored" in the phicon. The gesture can be thought of as "throwing" the cut region away.
- **Quit**: The user makes a mark like an "X" when they wish to stop the gesture recognition function.
- **Stationary**: In addition to the gesture models we also represent when the phicon is stationary.

To construct models for the gestures, each gesture was performed approximately half a dozen times and the trajectories were saved. For a given gesture the training trajectories were manually aligned and the mean trajectories were computed. A standard deviation from the mean trajectory was also computed for each curve. The trajectory models for each gesture are shown in Figure 4. The initial alignment of the curves could be performed using DTW. In addition to computing just the mean curve, we could compute "EigenCurves" as in [22].

In our experiments we take $\mu_{\max} = 9$ (there are nine primitive models), $\alpha_{\max} = 1.3$, $\alpha_{\min} = 0.7$ (we allow a 30% scaling), $\rho_{\max} = 1.3$, $\rho_{\min} = 0.7$ (a 30% temporal scaling). The standard deviations, $\sigma_i$, for the model trajectories were

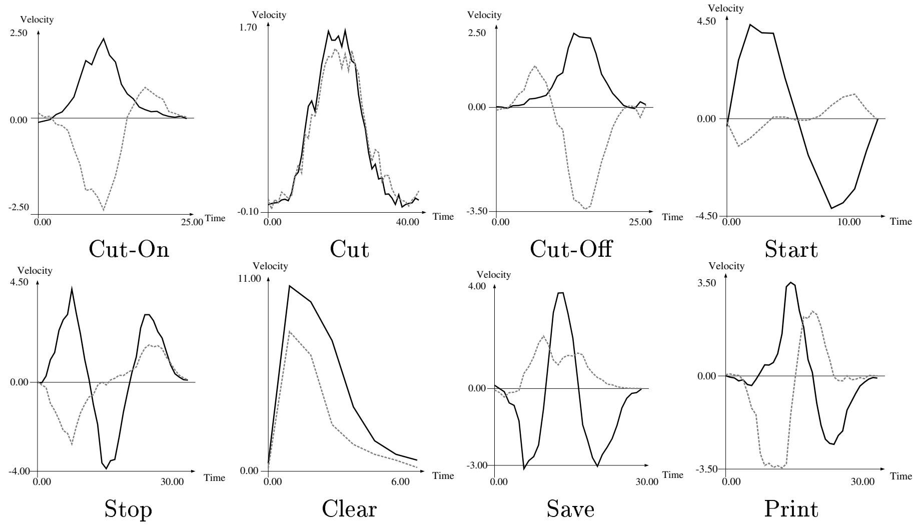


**Fig. 4.** Gesture Models. Temporal trajectories of horizontal (solid) and vertical (broken) velocity.

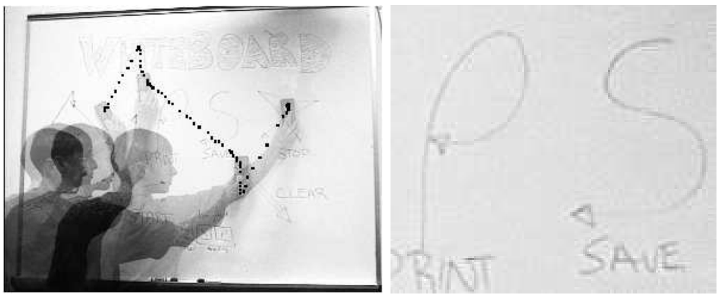

**Fig. 5.** Example of a "Cut" gesture. The user makes a gesture with the phicon. The image on the right is the region that is cut out of the larger image.

taken to be 1.0. Finally, the diffusion parameters were taken to be $\sigma_\rho = 0.01$ $\sigma_\phi = 0.05$ $\sigma_\alpha = 0.01$, and the temporal window was $w = 10$.

Figure 5 illustrates the performance of a "cut" gesture. We use a bright red block as our gesture phicon. The black dots in the figure represent tracked locations of the phicon. The image on the right is the region that is cut out by our algorithm.

Figure 6$a$ shows the horizontal and vertical velocities of the phicon as a function of time. This is our input data to which we will incrementally match all our gesture models.

The estimated value of the horizontal and vertical velocity is taken to be

$$\mathbf{u}_t = \sum_{n=1}^{S} \pi_t^{(n)} (\alpha \mathbf{m}_\phi^{(\mu)})$$

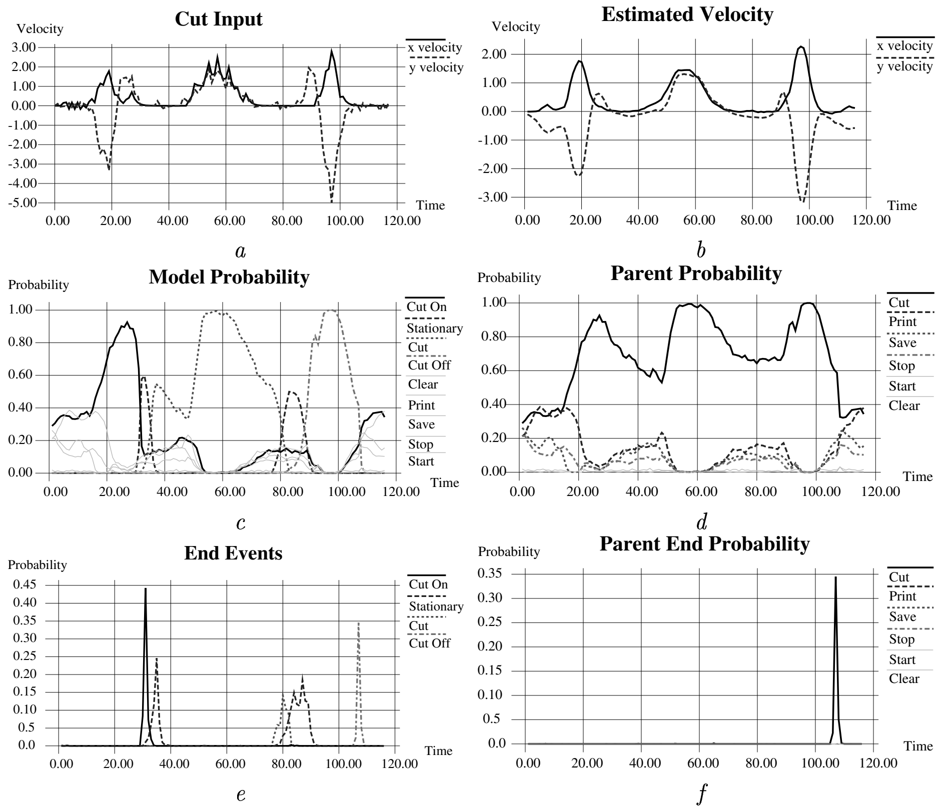


**Fig. 6.** "Cut" gesture. *(a),* Input horizontal and vertical velocity; *(b),* Estimated horizontal and vertical velocity; *(c),* Probability of each model type; *(d),* Probability of each parent gesture type; *(e),* Probability that the current state is a model completion event; *(f),* Probability that the current state is the completion of a parent.

where $\alpha \mathbf{m}_{\phi}^{(\mu)}$ is the estimated velocity for sample state $\mathbf{s}_t^{(n)}$. This represents the fit of the models to the data and is shown in Figure 6*b*.

Figure 6*c* shows the probability of each individual model as a function of time. Similarly Figure 6*d* shows the probability for the composite, parent, gestures. The probability of a particular model, $\mu$, is taken to be the sum of the normalized probabilities $\pi_t^{(n)}$ for which $\mu \in \mathbf{s}_t^{(n)}$.

Figures 6*e* and *f* show the probability that the trajectory models or parent gestures have completed respectively. This probability is given by

$$p(\mu^*) = \sum_{n=1}^{S} \begin{cases} \pi_t^{(n)} & \text{if } \mu \in \mathbf{s}_t^{(n)} \text{ and } \phi + 1 > \phi_{\max} \\ 0 & \text{otherwise} \end{cases}$$

where a sample is considered completed if the estimated phase parameter, $\phi$, is within one time instant of the maximum phase for that model/parent. The figures show clear spikes when the individual trajectory models ("Cut On", "Cut,"

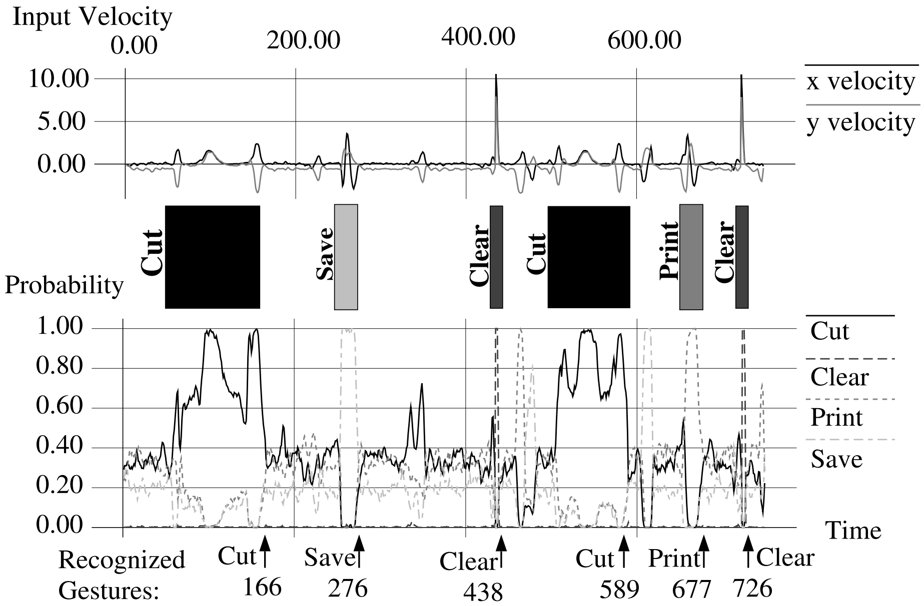


**Fig. 7.** Multiple-gesture experiment. *top:* Input horizontal and vertical velocity; *bottom:* Probability of each parent gesture type. The frame number at which the recognized gesture completes is given.

and "Cut Off") end. Similarly there is a spike when the parent cut gesture ends. If $p(\mu^*) > 0.1$ we consider $\mu$ to be recognized.

To actually cut the region (Figure 5) from the original image we must carry along with each state the time at which each trajectory model ended. This information is used at the end of the gesture to determine where the stationary mark points are located. The enclosed region of pixels is then extracted.

### 4.1 Multiple Gesture Experiment

We consider one more experiment that involves a series of gestures

Cut – Save – Clear – Cut – Print – Clear.

The input curves represent 850 samples of the horizontal an vertical velocities of the phicon (Figure 7 top). In this sequence, in addition to the actual gestures, the user moves the phicon between the gestures.

There are nine possible model types and six possible gestures. The input data at every frame is matched against these models and the data must be explained by some model. The normalized probabilities in Figure 7 (bottom) show that some of the non-gesture motions receive high normalized probabilities as we would expect. But using the probability for those gestures that actually are completed ($p(\mu^*) > 0.1$) we successfully recognize the completion of the gestures as shown along the bottom of Figure 7.

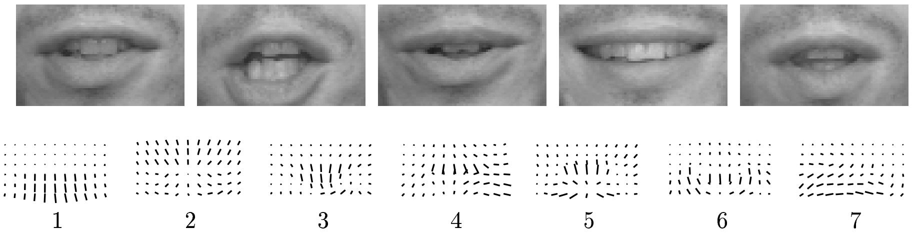


**Fig. 8.** Top: Example frames from the 600 image training set. Bottom: Basis flows for non-rigid mouth motion.

## 5 Facial Expressions

Parameterized models of optical flow provide a concise description of the image motion within a region in terms of a small number of parameters [2]. The evolution of these parameters over time can be used for motion-based recognition of facial expressions [3]. While here we consider optical flow, other authors have explored related approaches using deformable contours [1, 5, 12, 13, 15].

The horizontal and vertical image motion $\mathbf{u}(\mathbf{x}; \mathbf{a}) = (u(x,y), v(x,y))$ at a point $\mathbf{x} = (x, y)$ is represented as a linear combination of orthogonal basis flow fields, $M_i$

$$\mathbf{u}(\mathbf{x}; \mathbf{a}) \;=\; \sum_{i=1}^{k} a_i M_i(\mathbf{x}) \tag{3}$$

where $\mathbf{a} = (a_1, a_2, \ldots, a_k)^T$ are the coefficients of the model to be estimated.

Consider the example mouth images in Figure 8. To represent non-rigid motion of mouths we learn a parameterized model from example flow fields using principal component analysis (see [4]). Such a learned flow basis set is shown in Figure 8 for a training set that contained a variety of speech, a single smile, and four utterances of a test word. Since the image motion of the mouth is highly constrained, the optical flow structure can be modeled by a small number of principal component flow fields; in this case a seven parameter model is sufficient to account for 90% of the variance in the training data.

To recover the parameters we formulate an objective function to be minimized, namely

$$E(\mathbf{a}) = \sum_{\mathbf{x} \in R} \rho(I(\mathbf{x} + \mathbf{u}(\mathbf{x}; \mathbf{a}), t+1) - I(\mathbf{x}, t),\ \sigma), \tag{4}$$

where $\rho$ is a robust error function, $R$ is a set of pixels in an image region, and $I(\mathbf{x}, t)$ is the image brightness at pixel $\mathbf{x}$ and time $t$. The motion coefficients are estimated between frames using a standard regression-based optical flow algorithm [4].

The learned model is used to estimate the motion in a 150-image test sequence in which the subject smiles and speaks a word that occured in the training set.

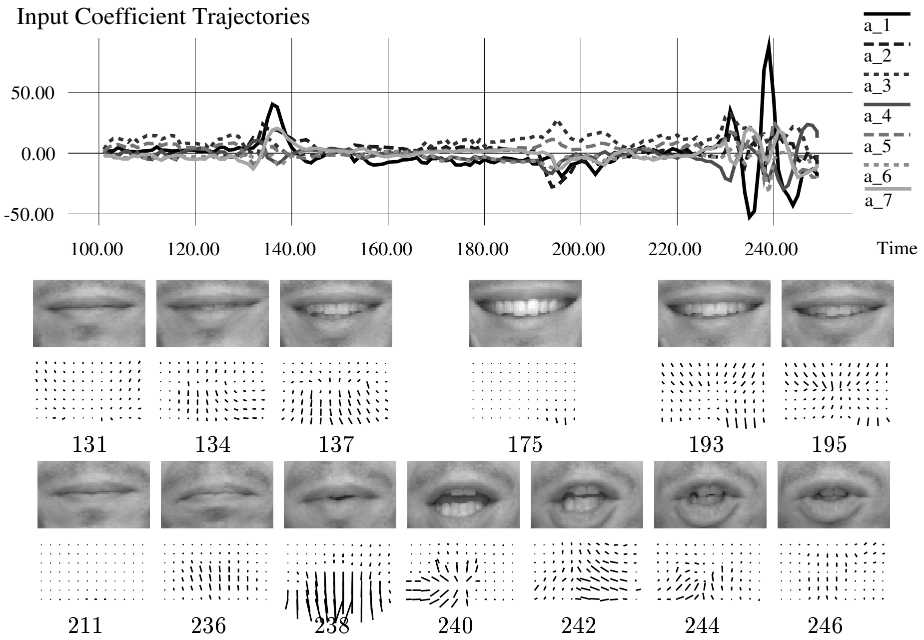


**Fig. 9.** Mouth motion experiment. Top: Recovered coefficients for the entire sequence. Below: Selected images and the corresponding estimated flow field are shown. Numbers under the images and flow fields correspond to frame numbers on the graphs.

A sample of the images from the sequence are shown in Fig. 9. Below each image is the estimated flow using the learned 7-parameter model.

The flow coefficients for the training sequence were computed and used to generate the trajectory models. The utterances of the test word were manually aligned and averaged to produce the trajectory models in Figure 10. Similarly, the single training smile was used to construct a compound model of a smile that is composed of primitive "onset," "apex," and "ending" models (cf. [3]). The apex is modeled as zero motion (all parameters zero) and may be of a prespecified or arbitrary duration. Given the small training set, we manually set $\sigma$ to 20.0.

Figure 11 shows the results of the CTR algorithm applied to the test data. Figure 11 (top) shows the probability of each of the trajectory models (Constant (apex), Smile-Start (onset), Smile-Stop (ending), and Utterance) over the sequence. Figure 11 (middle) shows the probability of the composite parent models (Smile and Utterance). Finally Figure 11 (bottom) shows the estimated probability that each parent model has completed. The completion of the smile expression and the utterance are readily detected though the ending of the smile is located less precisely. As in the previous section a probability of greater than 0.1 is a reliable indicator of the end of an expression.

This simple, preliminary, experiment illustrates how the CTR algorithm can be used for on-line expression recognition with multi-variate optical flow data.

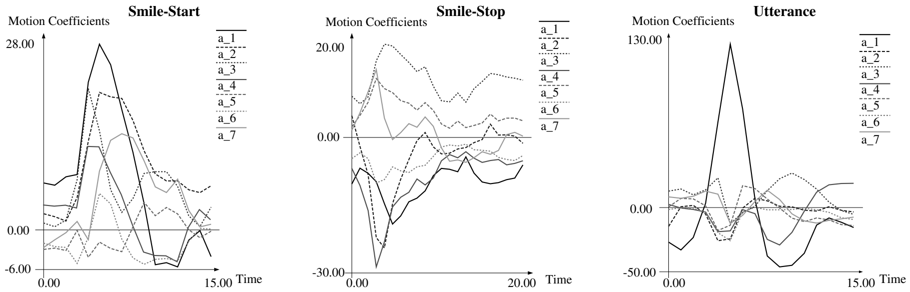


**Fig. 10.** Mouth expression models.

Our current research is exploring recognition with a richer set of facial expressions and speech acts. In particular, we are defining a vision-based user interface that uses mouth motions to control various computer screen functions.

## 6 Discussion and Conclusions

We have described an on-line, probabilistic, framework for matching temporal trajectories using the CONDENSATION algorithm. The framework extends previous work on CONDENSATION tracking to the problem of recognizing gestures based on stored temporal models. We have demonstrated how such a method can be used to recognize simple gestures and facial motions. In particular, we developed an augmented whiteboard application that allows users to interact with a whiteboard "scanner."

Note that in previous work [8, 9], the CONDENSATION algorithm has been used to track objects with learned spatial and temporal dynamics. In this paper the tracking, or motion estimation, problem was solved separately and the CONDENSATION algorithm was used only to perform recognition given the temporal trajectories. In future research we will explore the integration of the motion estimation and recognition problems by using the probability distribution over states to help constrain the estimation problem. Note that for optical flow estimation this may prove more difficult than for the tracking applications in [9].

In the experiments presented here we took $S = 1000$ samples and the resulting experiments run significantly slower than real time. Blake and Isard have demonstrated real-time versions of a similar algorithm which suggests that our method might be suitable for real-time recognition (with appropriate optimizations). In particular, real-time performance is achieved when only 100 samples are used. The number of samples required for a particular problem is dependent on the number of models. We achieve real-time performance with the mouth example using only 100 samples with no reduction in accuracy. The gesture example, on the other hand, has a richer model-base and, hence, requires more samples.

In our current work, segmented and aligned training data was provided and transition probabilities between events in the parent models were set by hand.

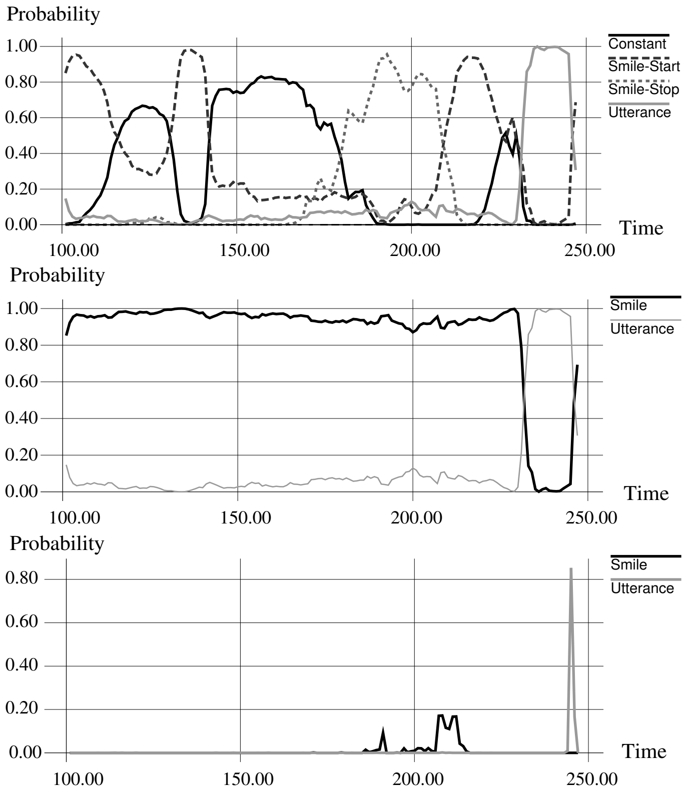


**Fig. 11.** Mouth expression recognition results.

While the method has very few parameters, it would be worth exploring how the techniques used for training HMM's might be used to generate the models automatically. Additionally, the size of the temporal window $w$ might be learned from the training data.

In this paper we have explored the application of the CTR framework to the recognition of human motion (gestures and facial motions). The system could also be used to recognize other sorts of gestures as well as speech and on-line handwriting. In summary, random sampling techniques provide an interesting new framework for the automatic analysis of human motion.

**Acknowledgements.** We thank Francois Bérard for providing the real-time phicon tracking and Gudrun Socher for discussions about whiteboards and gesture interfaces.

## References

1. A. Baumberg and D. Hogg. Learning flexible models from image sequences. *ECCV-94*, pp. 299–308, Stockholm, 1994.

2. M. J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *CVIU*, 63(1):75–104, Jan. 1996.
3. M. J. Black and Y. Yacoob. Recognizing facial expressions in image sequences using local parameterized models of image motion. *IJCV*, 25(1):23–48, 1997.
4. M. J. Black, Y. Yacoob, A. D. Jepson, and D. J. Fleet. Learning parameterized models of image motion. *CVPR-97*, pp. 561–567, Puerto Rico, June 1997.
5. A. Blake, M. Isard, and D. Reynard. Learning to track the visual motions of contours. *Artificial Intelligence*, 78:101–134, 1995.
6. C. Bregler. Learning and recognizing human dynamics in video sequences. *CVPR-97*, pp. 568–574, Puerto Rico, June 1997.
7. J. L. Crowley and F. Berard, Multi-Modal Tracking of Faces for Video Communications, CVPR'97, pp. 640–645, Puerto Rico, June 1997.
8. M. Isard and A. Blake. Contour tracking by sotchastic propagation of conditional density. *ECCV-96*, pp. 343–356, Cambridge, UK, 1996.
9. M. Isard and A. Blake. A mixed-state Condensation tracker with automatic model-switching. *ICCV'98*, pp. 107–112, Mumbai, India, Jan. 1998.
10. H. Ishii and B. Ullmer. Tangible bits: Towards seamless interfaces between people, bits and atoms. *Proc. of CHI'97*, 1997.
11. K. Kanazawa, D. Koller, and S. Russell. Stochastic simulation algorithms for dynamic probabilistic networks. *Proc. of the Eleventh Conf. on Uncertainty in AI*, Montreal, 1995.
12. R. Kaucic, B. Dalton, and A. Blake. Real-time lip tracking for audio-visual speech recognition applications. *ECCV-96*, pp. 376–387, Cambridge, UK, 1996.
13. A. Lanitis, C. J. Taylor, T. F. Cootes, and T. Ahmed. Automatic interpretation of human faces and hand gestures using flexible models. *Int. Conf. on Automatic Face and Gesture Recognition*, pp. 98–103, Zurich, 1995.
14. L. R. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Readings in Speech Recognition*, 1989.
15. D. Reynard, A. Wildenberg, A. Blake, and J. Marchant. Learning dynamics of complex motion from image sequences. *ECCV-96*, pp. 357–368, Cambridge, UK, 1996.
16. D. Sankoff and Eds. J. B. Kruskal. *Time warps, string edits, and macromolecules: The theory and practice of sequence compression*. Addison-Wesley Pub., Reading, Mass., 1983.
17. Q. Stafford-Fraser. Brightboard: A video-augmented environment. *Proc. of CHI'96*, 1996.
18. T. Starner and A. Pentland. Visual recognition of American Sign Language using Hidden Markov Models. *Int. Conf. on Automatic Face and Gesture Recognition*, pp. 189–194, Zurich, 1995.
19. R. Szeliski. Image mosaicing for tele-reality applications. *Second IEEE Workshop on Applications of Computer Vision*, pp. 44–53, Sarasota, Florida, 1994.
20. A. D. Wilson, A. F. Bobick, and J. Cassell. Temporal classification of natural gesture and application to video coding. *CVPR-97*, pp. 948–954, Puerto Rico, June 1997.
21. A. D. Wilson and A. F. Bobick. Recognition and interpretation of parametric gesture. *ICCV-98*, pp. 329–336, Mumbai, India, Jan. 1998.
22. Y. Yacoob and M. J. Black. Parameterized modeling and recognition of activities. *ICCV'98*, pp. 120–127, Mumbai, India, Jan. 1998.
23. Y. Yacoob and L. Davis. Parameterized modeling and recognition of activities. *ICCV'98*, pp. 446–453, Mumbai, India, Jan. 1998.