

# Two Methods of Gesture Recognition

Alexander Gruenstein

March 4, 2002

## 1 Introduction

In this paper I will review two recent approaches to gesture recognition. First, I will describe the Condensation algorithm [2] and its extensions to gesture recognition [3] [1]. Then, I will describe a method that uses multi-scale motion segmentation to monitor motion over time and a Time Delayed Neural Network to match this motion to gestures [4]. My focus will be on how these algorithms use models of motion over time to classify gestures. Finally, I'll summarize a few of the comparative advantages of each algorithm.

## 2 The Condensation Algorithm and Gesture Recognition

The Condensation algorithm (Conditional Density Propagation over time) [2] was originally proposed to deal with the problem of tracking curves in clutter. It makes use of random sampling in order to model arbitrarily complex probability density functions. That is, rather than attempting to fit a specific equation to observed data, it uses  $N$  weighted samples to approximate the curve described by the data. Each sample consists of a *state* and a *weight* proportional to the probability that the state is predicted by the input data. As the number of samples increases, the precision with which the samples model the observed pdf increases. When applied to tracking, a *state* corresponds to parameters representing the state of the object being tracked – for instance, its velocity and location.

The real utility of this representation becomes evident as observations are made over a series of time steps  $t_1, \dots, t_n$ . In order to generate the new sample set at time  $t+1$ , states are randomly selected (with replacement) from the sample set at  $t$ , based on their weight; that is, the weight of each sample determines the probability it will be chosen. Given such a randomly sampled state  $\mathbf{s}_t$ , a prediction of a new state  $\mathbf{s}_{t+1}$  at time step  $t + 1$  is made based on a predictive model. This corresponds to sampling from the process density  $p(\mathbf{X}_{t+1} | \mathbf{X}_t = \mathbf{s}_t)$ ,

where  $\mathbf{X}_t$  is a vector of parameters describing the object's state. Finally,  $\mathbf{s}_{t+1}$  is assigned a weight proportional to the probability  $p(\mathbf{z}_{t+1}|\mathbf{s}_{t+1})$ , where  $\mathbf{z}_{t+1}$  is a set of parameters describing the observed state of the object at time  $t + 1$ . In this way, predicted states that correspond better to the data receive larger weights. Since arbitrarily complex pdfs can be modeled, an arbitrary number of competing hypotheses (assuming sufficiently large  $N$ ) can be maintained until a single hypothesis dominates.

Clearly a critical part of the algorithm is choosing an appropriate prediction function that predicts a state  $\mathbf{s}_{t+1}$  from a randomly sampled state  $\mathbf{s}_t$ . The better the underlying model, the better the predictions will align with the observations. In [3] an extension of the basic Condensation algorithm that automatically switches between various prediction models (in this case, motion models) is described. This has two significant advantages. First, the performance of the tracker improves because multiple models allow the prediction of different types of motion. Second, and more relevant to gesture recognition, is that the likelihood that a particular model is the correct model for the motion observed at a given time  $t$  can be extracted. If each competing model represents a single gesture, then the most likely model predicts which gesture is being observed.

In order to implement the mixed-state model, more content must be added to each *state*. States are now defined as:

$$\mathbf{X} = (\mathbf{x}, y), \mathbf{x} \in \mathbb{R}^{N_M}, y \in \{1, \dots, N_S\}$$

where  $y$  labels the current model, and  $\mathbf{x}$  is the content of our original states – namely, a vector describing the state of the object being tracked. We can then predict the probability of a new state at time  $t + 1$  given a state at time  $t$  as:

$$p(\mathbf{X}_{t+1}|\mathbf{X}_t) = p(\mathbf{x}_{t+1}|y_{t+1}, \mathbf{X}_t)p(y_{t+1}|\mathbf{X}_t)$$

where we define transition probabilities between each pair of states like so:

$$T_{ij}(\mathbf{x}_t) = p(y_{t+1}|\mathbf{X}_t) = p(y_{t+1} = j|\mathbf{x}_t, y_t = i)$$

Essentially, given a previous model and object state, we predict a new model; then, based on that model, we predict a new object state. In [3] a two-state model is applied to a bouncing ball (one model for pre-bounce motion and one for post-bounce motion). A three-state model is applied to track simple hand gestures: it is used to track a hand drawing with a pen where the hand can be drawing lines, stationary, or scribbling.

In [1] the mixed-state Condensation algorithm is extended to recognize a greater number of gestures. Gestures are recognized based on their *temporal trajectories*. The authors focus solely on applying Condensation to identifying temporal trajectories, and leave for future work the problem of leveraging the temporal models as constraints on object tracking. As such, they ignore the tracking aspect of the problem all together: only simple *phicons* – physical-object icons – of distinctive color are tracked by a color blob tracker.

The predictive models used in this application of Condensation are discretely-sampled curves that represent the trajectory of the phicon as it moves through

a given gesture. The trajectory is represented by plotting both the horizontal and vertical velocities of the phicon against a phase parameter  $\phi$ . The model for each gesture was computed based on approximately six training sequences; the mean trajectory and standard deviation were computed.

In this application of Condensation, a *state* at time  $t$  is described as a parameter vector:  $\mathbf{s}_t = (\mu, \phi, \alpha, \rho)$  where:

$\mu$  is the integer index of the predictive model;  
 $\phi$  indicates the current position in the model;  
 $\alpha$  refers to an amplitudal scaling factor;  
 $\rho$  is a scale factor in the time dimension.

The sample set is initialized with  $N$  samples uniformly distributed among the possible states. In the prediction step, each parameter of a randomly sampled  $\mathbf{s}_t$  is used to determine  $\mathbf{s}_{t+1}$  based on the model parameter ( $\phi_t$ ) of that particular  $\mathbf{s}_t$ . The weight of  $\mathbf{s}_{t+1}$  is calculated based on how well the observed trajectory matches the model's trajectory (parameterized according to the parameter vector in  $\mathbf{s}_{t+1}$ ) over a historical time window  $w$ . At a given time  $t$ , we can determine the most likely gesture being observed by the system by finding the model with the most cumulative weight. A series of gestures can be recognized by introducing transition probabilities between models and choosing a threshold for  $p(\mu^*)$  above which  $\mu$  is classified as recognized.

This is a very interesting application of Condensation to the problem of matching temporal trajectories for gesture recognition. While it works in the small domain described in the paper, it would be interesting to see how well it would scale up. Only a handful of 'whiteboard' gestures (for example: Save, Cut, Erase) can be recognized, and the gestures seem to be intentionally chosen to be extremely distinct from one another. In scaling this system up to a larger set of gestures (for instance American Sign Language), it would be interesting to see what other sorts of *a priori* constraints could be added to increase recognition accuracy. For instance, ASL, like all natural languages, is constrained in its order by a grammar – only certain combinations of signs are allowed. Factoring in this sort of *a priori* information – either by modifying the criterion for determining that a particular gesture has been *recognized* or possibly by biasing the initialization of the set by using a non-uniform distribution – might increase recognition performance. Additionally, dynamically adjusting the number of samples in the sample set (by decreasing the number of samples needed when one hypothesis is dominating) might increase recognition speed.

### 3 Motion Trajectories and TDNNs

In [4] the authors present a different algorithm for recognizing gestures (specifically, American Sign Language [ASL]). Where the previous papers worked towards using Condensation to both track the object of interest and match the trajectory of that object over time to a given model, in [4] distinct algorithms

are used for the two steps. The authors use multi-scale motion segmentation to track movement between frames and a Time Delayed Neural Network (TDNN) to match the trajectory of that motion to a given gesture model.

To track motion, sequential pairs of frames are analyzed to identify regions of motion. First, the image is segmented into a hierarchy of regions defined by a family of attraction force fields. Next, each region is matched between frames at four different scales and regions are matched using a dynamic programming graph algorithm. Then an affine transformation between each pair of matched regions is estimated at each of the four scales, using an iterative method. Next, the motion fields at each of the four scales are combined to form a single motion field. Finally, a heuristic is used to segment the resulting motion field into areas where the motion is uniform.

In order to recognize sign language, domain dependent assumptions about which areas of motion are of interest are used. Particularly, only areas of motion where skin color is detected are considered. Furthermore, signs are described by the relation between the head (approximated by a large ellipse), and either the palm (an ellipse) or the closed hand (approximated by a rectangle) and so regions of motion are merged until the shape of the merged region is either an ellipse or rectangle. The larger elliptical region is then identified as the head, and the next two smaller elliptical regions as palms (or, if rectangular, as closed hands). Clearly, this algorithm might run into difficulties in an environment with a noisy, human-skin-colored background (a scene, for instance, with other moving people in the background).

In order to classify the motion of the various regions over time as a particular sign in ASL, a Time Delay Neural Network with two hidden layers is employed. The structure of the TDNN is much like that of a regular neural network, except that at each layer there are input slots that correspond to a particular time window of input. At each layer from Input to Output, a progressively larger time window of inputs is inserted into each 'slot'; while there might be 50 slots in the Input layer (one slot for each window of length 5), there are only 46 slots in the first hidden layer (and hence a longer window length), and progressively fewer slots in each of the following layers. The output layer has one slot for each motion trajectory model. The inputs to the network are vectors of  $(x, y, v, \theta)$  where  $x$  and  $y$  are positions relative to the center of the head,  $v$  is the magnitude of the velocity, and  $\theta$  is its angle. The TDNN is trained with backpropagation.

After training on a database of 40 ASL signs, with about 38 instances of each sign, a recognition rate of 96.21% on novel data was achieved.

## 4 A Brief Comparison

The two algorithms for gesture recognition presented here are quite different. The TDNN method has certainly been shown to be effective for a medium size vocabulary of gestures (in this case, ASL signs) while the Condensation method has only been shown to be effective on a relatively small vocabulary of simpler, artificially designed gestures. Furthermore, even for this small vocabulary of

artificial signs, the Condensation algorithm uses only simple color-blob tracking instead of a more robust tracker.

#### 4.1 Advantages of TDNN/segmentation

There are a couple of advantages of the TDNN/segmentation method:

1. A well-developed training method exists for neural networks (back propagation) and has been used to effectively train the TDNN in the algorithm surveyed here with only labeled training data. In contrast, developing the temporal trajectories for the Condensation algorithm is a relatively *ad-hoc* process that may have difficulties capturing the way that different individuals make the same gesture.
2. The TDNN/segmentation method takes both hands, instead of just a single hand, as input; hence, the gestures it can recognize are more complex.

#### 4.2 Advantages of Condensation

1. Because Condensation can be used for both tracking of the object in question and temporal-spatial classification, there is a potential that the temporal projection models can act as constraints to make the tracking problem more robust.
2. Condensation has been shown to work in real time under certain conditions. It is unclear how fast the TDNN/segmentation algorithm runs.
3. Condensation has been adapted recognize sequences of gestures, whereas the TDNN/segmentation algorithm only recognizes a single gesture in isolation.

## References

- [1] Michael J. Black and Allan D. Jepson. A probabilistic framework for matching temporal trajectories: Condensation-based recognition of gestures and expressions. In *Proceedings 5th European Conference Computer Vision*, volume 1, pages 909–924, 1998.
- [2] Michael Isard and Andrew Blake. Contour tracking by stochastic propagation of conditional density. In *Proceedings European Conference on Computer Vision*, pages 343–356, 1996.
- [3] Michael Isard and Andrew Blake. A mixed-state condensation tracker with automatic model-switching. In *Proceedings 6th International Conference Computer Vision*, pages 107–112, 1998.
- [4] Ming-Hsuan Yang and Narendra Ahuja. Recognizing hand gesture using motion trajectories. In *IEEE CS Conference on Computer Vision and Pattern Recognition*, volume 1, pages 466–472, June 1999.