Real-Time Tracking for Visual Interface Applications in Cluttered and Occluding Situations

David Bullock, John Zelek
School of Engineering, University of Guelph
Guelph, Ontario, Canada, N1G 2W1
dbullock/jzelek@uoguelph.ca

Abstract

Visual interface systems require object tracking techniques with real-time performance for ubiquitous interaction. A probabilistic framework for a visual tracking system which robustly tracks targets in real-time using color and motion cues is presented. The algorithm is based on particle filtering techniques of the I-Condensation filter. An innovation of the paper is the use of motion cues to guide the propagation of particle samples which are being evaluated using color cues. This results in a probabilistic blob tracking method which is shown to greatly outperform conventional blob trackers when in the presence of occlusion and clutter. A second innovation presented is the use of motion-based temporal signatures for the visual recognition of an initialization cue. This allows for passive initialization of the tracking system. The application presented here is the task of digital video annotation using a hand-held marking device.

1 Introduction

Computer vision technologies offer a natural medium for human-computer interfaces. People naturally tend to express themselves using gestures, expressions, and actions. There have been extensive research initiatives addressing the problem of real-time tracking for interface purposes. However, most solutions have claimed success by employing one of several simplification techniques: using a constrained and uncluttered environment [1], relaxing the real-time constraint [2], or assuming that minimal occlusion will occur [3, 4]. Unless highly constrained, all visual interfaces will inevitably be faced with ambiguous data. This ambiguity can result from the tracked object being temporarily occluded, agile target movement, or the target becoming camouflaged by clutter that is similar in appearance. A visual interface system must have the ability to identify ambiguous scenarios, and be able to re-acquire the true target once it becomes visible again. Our system is able to maintain robustness even in the presence of ambiguity by employing two powerful tracking techniques. First, particle filtering is used which represents the target posterior distribution as a collection of weighted samples in state space. This is a natural mechanism for maintaining multiple modes (hypotheses) and propagating the uncertainty over time. This is a strength of the Condensation algorithm [5] compared to conventional predictive filters such as the Kalman filter [6] which are limited to uni-modal Gaussian posterior representations. Secondly, our system employs a motion detection algorithm and uses it to redistribute a portion of the samples into high-probability regions of state space using importance sampling [7]. The sample weight represents the likelihood of a particular sample being the true target location and is calculated by comparing the image data to the a priori target color model. The motion information is not used to gage the likelihood of a sample, but rather it only identifies areas in the state-space that should be investigated by samples. This is an effective technique as targets are most commonly lost during periods of movement, and this method allows the system to investigate these regions of movement in a principled Bayesian manner.

A second innovation of this paper is the implementation of a passive initialization scheme for activating the interface device. Vision-based interface devices are typically employed in office or classroom environments in which there may exist extended periods of time between device usages. During these periods of inactivity it is inappropriate to keep the system in the continuous tracking mode, as this can result in temporary misclassification of an individual simply passing through the field-of-view as being a target user. Therefore, a method of initialization and activating the system is required. Ideally this method should not require the explicit intervention of the user by having them hit an activation key. Rather, a passive initialization strategy is presented in this paper in which target individuals are recognized by having them perform a simple pre-defined visual initialization cue which the system is tuned to detect via template

matching of temporal motion signatures.

The application presented here is the digital annotation of video sequences. This is achieved through the real-time visual tracking of a hand-held marker object, in this case, a green ball. The user is able to annotate over top of a video stream by moving the ball as they would a marker on a piece of paper. By intentionally hiding the ball within their hand, the user is able to temporarily halt the annotation process, thus allowing for "pen-up" and "pen-down" capabilities. Our method is also capable of advanced marking features such as altering the appearance of the annotations based on the target object velocity. Video clips demonstrating the tracking technique are accessible at http://www.uoguelph.ca/ dbullock/trackdemo.htm.

2 Initialization

Before target tracking can begin, the system must recognize that an eligible target is in the field-of-view. Since most vision-based interface devices will operate in office or classroom environments, there must exist a means for differentiating between individuals simply passing through the camera's field-of-view, and those who wish to use the system as an interface tool. This is an initialization issue for which three general solutions exist.

1. No Initialization: The tracking system is "on" at all times and attempting to localize the target in the field-of-view. This is problematic, particularly in systems which rely on color-based tracking cues, as an individual passing through may be momentarily mistaken for a target when the device is not in use.

2. Active Initialization: In this scenario, the tracking system is explicitly told that an individual is present and wishes to use the interface device. Most often this takes the form of the user hitting an activation button.

3. Passive Initialization: In these systems the tracking algorithm is activated by the system recognizing that an individual is present and attempting to interface with the device. Target recognition is achieved by visual recognition of a pre-defined initialization cue. This results in a more transparent and easy-to-use device than method (2).

The passive initialization cue used here is to have the user hold the interface object (the colored ball) in front of them while moving it in a circular pattern. Once this initialization cue is recognized by the system, tracking can begin as outlined in section 4. Recognition of initialization cues is done by template matching of temporal motion signatures, as will be discussed in the following section.

2.1 Motion History Images

Motion history images (MHI) as proposed in [8] are a means of representing spatial and motion information in the form of an image. The brightness of a pixel is determined by how recently there has been motion detected at that location. As shown in figure 1, pixels currently in motion appear white. Pixels in various shades of gray represent locations which have detected movement in the recent past. The MHI shown in figure 1 shows a typical motion history image generated from the pre-defined initialization cue used in our experiments. This particular motion cue was selected

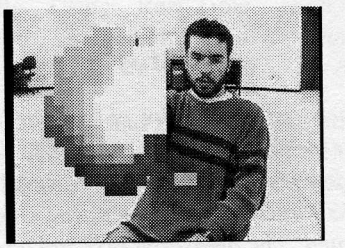

Figure 1: *The motion history image represents how recently motion was detected at each pixel location. Pixels currently in motion are shown as white, while pixels in motion in the recent past are shown in decaying shades of gray.*

as it produces a distinct MHI not usually generated by random movements of individuals. Movement is detected using Camus optical flow [9] which allows for real-time calculation of optical flow values. Please refer to section 4.3 for a discussion on the general problems associated with optical flow methods, a justification for our selection of the Camus method, and the deviations which we made from the original algorithm. The gradient of motion can be calculated from the MHI using a simple Sobel gradient mask for convolution. While motion vectors are generated by the optical flow algorithm itself, we desire a richer set of vectors spanning the region which has seen movement in the recent past.

2.2 Temporal Signatures

Having produced a measure of recent movement in the spatial domain, we require a means of comparing these MHI images to a known template for circular movement of a forward extended hand. We use a technique inspired by [10] which is scale and translation invariant in detecting a known action. The algorithm involves the definition of nine overlapping spatial windows (as defined in [10]), in each of which the distribution of motion vectors is calculated. In order to make the technique translation invariant, the windows are centered around the calculated centroid of motion pixels. In order to gain scale invariance, the windows are sized in order to cover the bounding box of recent motion. As is

done in [10] we use an angular resolution of $\pi/6$ radians (12 bins) in each histogram. Within each window, motion vectors are calculated at a sub-sampled resolution (we use a $\frac{1}{5}$ sub-sampling). The direction of the motion gradient is found, and the appropriate bin in the motion histogram is incremented. Once all of the windows have been processed, the nine histograms are placed end-to-end to create a single MHI motion histogram. Thus in our experiments, the final histogram consists of 12x9=108 bins. In order to ensure true scale invariance and target speed invariance, we normalize the histogram to consist of a total of 10,000 units distributed across all of the 108 bins. This final motion histogram is known as the temporal signature. The temporal signature corresponding to the circular movement of a forward extended hand (the initialization cue) is shown in figure 2.

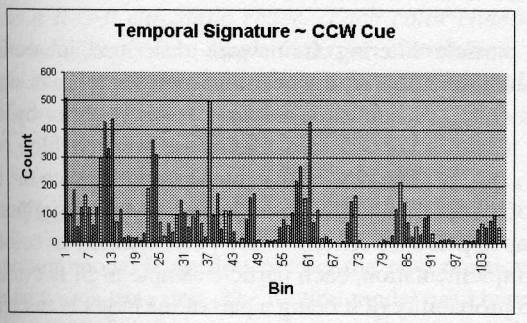

Figure 2: *Calculated Temporal Signature: The above signature corresponds to the MHI of the counter-clockwise circular movement of a forward extended hand. The signature is a means of representing the spatial distribution of motion vector orientations.*

2.3 Action Templates

In order to measure the likelihood of a particular action presently being performed, the calculated temporal signature is compared to a set of pre-defined temporal signatures of known actions. The actual comparison process is done via a simple sum-of-absolute-differences calculation. This produces a value which can be thresholded to give a likely/not-likely differentiation for an action being performed. The event detection threshold is determined empirically and is dependent of the initialization cue selected. For the purposes of our system, only two actions are of significance: counter-clockwise and clockwise circular hand movement. Building these action templates is done by taking a sample of 10 temporal signatures during a training phase. These temporal signatures are then averaged to create a single action template for each of the actions. This is done to reduce the effects of any statistical anomalies present in any one temporal signature. To account for users who perform the initialization cue at various speeds, a set of fast movements and a set of slow movements was captured for each of the two pre-defined actions. Thus, a total of four temporal signature templates are learned. Upon detection of a initialization cue, the system is activated and tracking begins as outlined in section 4.

3 Predictive Filtering

Computation of the target posterior can be a difficult endeavor due to non-linearities of the likelihood function over model parameters. These non-linearities result from occlusion, one:many matching ambiguities, and camouflaging clutter. Further complicating the representation of the posterior is the non-linear dynamics of the temporal prior. Thus, any assumption of uni-modality or Gaussian distributions in the posterior distribution is not appropriate [5]. This is the primary shortcoming of the Kalman filter. The need for a filter capable of multi-modal non-Gaussian distributions is the motivation for representing the target posterior by a set of weighted state space samples. These samples are then made to propagate through time via the motion (temporal) model and the observed image likelihoods. With a large enough set of samples, this weighted particle set can be a fair representation of the true target posterior. The seminal work on the use of particle sampling in visual tracking was presented as the Condensation algorithm [5]. The algorithm employs the Monte Carlo technique of factored sampling to propagate a set of samples through state space in an efficient manner. In our experiments, only the location of the target is tracked through time, and thus, the state, s_t, is represented by a 2x1 vector of parameter assignments, $s_t = [x_t, y_t]$. At time step t-1, the posterior is represented by N state samples, each corresponding to a candidate location for the target and weighted according to their relative likelihood. This likelihood is calculated by comparing the image data at the location of s_t with the a priori known color model of the target, as will be discussed in section 4. This algorithm makes use of Bayesian inference which allows for the inferring the posterior state density $p(X_t|Z_t)$ where X_t is the state and Z_t is the image data. The prior, $p(X_t|Z_{t-1})$ is inferred from predicting $p(X_{t-1}|Z_{t-1})$ though the motion model $p(X_t|X_{t-1})$, and this is used as a basis for calculating the measurements (observations) $p(Z_t|X_t)$, from which the posterior is computed. To ease the computational requirements, the posterior is only estimated up to an unknown scale factor α as shown in (2). See [5] for a more detailed explanation of this.

$$p(X_t|Z_t) = \frac{p(Z_t|X_t)p(X_t|X_{t-1})}{p(Z_t)} \quad (1)$$

$$= (\alpha)p(Z_t|X_t)p(X_t|X_{t-1}) \quad (2)$$

The schedule of events in the Condensation algorithm at each time step follows as shown below:

1. A new set of samples $S_t = [s_1, s_2, ..., s_N]$ is selected from S_{t-1} using a sample-and-replace scheme with preference given to samples with larger weightings.

2. Samples are propagated through state space using the motion model. Typically this includes a deterministic drift component and a random diffusion component as we present in section 4.2.

3. Each sample in S_t is then weighted using the likelihood function and the data from image I_t. This weighted set of samples now represents the approximation of the posterior at time t.

4. If discrete feedback is required the sample set S_t can be queried by one of many statistical operators including $max(S_t)$, $mean(S_t)$, $median(S_t)$, etc.

3.1 Importance Sampling

Under the standard implementation of the Condensation algorithm, the population from which the sampled set S_t is created, is the previous sample set S_{t-1}. Thus, the portions of state-space available for examination in a time-step are limited before any measurements are made. With a good approximation of the state distribution , and a large number of samples, this would not be problematic as even unlikely locations in state-space would be examined as the target state propagates over time. However, due to a finite number of samples being used, the vast majority of all samples are likely to be concentrated near the most likely object positions. Interestingly, this is both a potential major flaw of the algorithm, and the source of why it can represent a high dimensional state-space with a small number of samples [11]. This understanding prompted the development of the I-Condensation algorithm [7] which uses importance sampling to direct the search and redistribute particles from low-probability areas to high-probability regions. Over time the output posterior distribution of the algorithm may no longer accurately approximate the true state distribution. This is particularly true in the case of temporary occlusion of the target. In such situations some samples are repositioned into an area of the state-space that houses higher target probabilities. Doing this requires that we use additional knowledge of the targets whereabouts, beyond that of the posterior generated by S_{t-1}. This is known as the Monte Carlo technique of Importance Sampling [12]. Typically an importance sampling function represents a second tracking modality (image cue), which complements the primary tracking modality by identifying potential targets that the primary modality may have missed. Most importantly, it can aid the primary tracking modality by focusing in on regions of high probability for the target. The importance sampling function does not influence the calculation of sample likelihoods. Instead, it merely aids in positioning the samples into areas of state-space that may be high probability regions. In [7], importance sampling is used by having a blob tracker identify skin colored regions, and then feed this information to a high level contour tracker which does hand tracking. The conventional Bayesian approach to fusion of multiple image cues would be to combine measurements from the various modalities, and weight them according to their inverse variances [7]. However, this is problematic as it assumes the statistical dependencies of the image cues are known and independent, which is rarely the case.

4 Method

The particle filtering framework described in section 3 was implemented as a probabilistic blob tracker in our experiments. A conventional blob tracker works by scanning a search window centered at the previous blob center (x_{t-1}, y_{t-1}). The new blob center location is then the average of all pixel co-ordinates in the search window which satisfy an appearance criteria (often pixel intensity or color). In our implementation, each particle sample in S_t is evaluated and a probability of it being a part of the target is calculated. The probabilities are then used to assign the weightings to each sample particle. There is no need for a normalization constant as the relative value of the weightings are all that are of significance. A key difference between a conventional blob tracker and our probabilistic particle-based scheme, is that ours has no explicit notion of the target location. Rather, the target location can only be determined by evaluating one of the statistical properties of the posterior (mean, median, mode, etc.). To allow for visual feedback annotated on top of the video stream, the mean of S_t is taken to be an approximation of the target location at time t.

4.1 Observational Model

The observational model calculates the $p(Z_t|X_t)$ likelihoods required for the sample weighting process in step (3) of the algorithm in section 3. The observational model used is a probabilistic color-space definition. This is a color model of the target known a priori and is the primary means for segmenting the target from the background and clutter. The color model makes the assumption that the color space of the target can be approximated by three Gaussian distributions. The color model is defined by 6 parameters: a mean and variance define the observational likelihood on each of the 3 color channels (R,G,B). This results in a likelihood function which returns a value in the range [0,1] given the

pixel R,G,B values. Figure 3 illustrates how the three Gaussian distributions are used to calculate the fitness of specific RGB values. In the example the RGB values (r=3, b=14, g=30) produce a likelihood value of 0.373. The 6 values in the color model are determined prior to running the tracker by sampling a number of pixels within an area known to house the target.

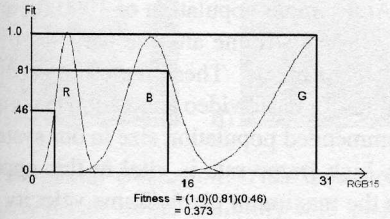

Figure 3: *The observational model is defined probabilistically in a R-G-B chromatic space. Each color channel is defined individually by a single Gaussian distribution. The total fitness of a color value is the product of the individual distribution scores, as shown above.*

4.2 The Motion Model

The motion model is used to predict the new positions of samples at each time step. The motion model used here is a simple one which is based on a deterministic drift and random diffusion component. The deterministic drift component represents the assumption of constant velocity. The random diffusion component represents the target's tendency to make sudden changes in direction and velocity (as is often seen with hand gestures). The propagation of state variables via the motion model is defined by equation (3):

$$p(s_{i,t}|s_{(i,t-1)}) = G(s_t - (s_{t-1} + v_{i,t-1}, \sigma) \qquad (3)$$

where $v_{i,t-1}$ denotes the sample velocity in pixels/frame of s_i, and $G(x, \sigma)$ denotes a Gaussian distribution with zero mean and standard deviation σ, evaluated at x. The value of σ is dependent on the physical characteristics of the target and camera/target layout and is determined empirically.

4.3 Using Motion Cues to Direct the Search

The use of importance sampling is a deviation from step (1) in the algorithm in section 3. A key innovation of this paper is the use of motion cues to redistribute particle samples which are being evaluated using color cues. The motion information would not be appropriate for use as the primary tracking means since the target may remain still for large periods of time. However, during periods of motion, it acts as a valuable means of directing the search. Importance sampling [12] is a Monte Carlo technique which we use for both initialization and re-acquisition of lost targets in this experiment. Due to the tendency of targets to move suddenly and rapidly in interface applications, the ability to quickly reacquire the target is of paramount importance. The importance sampling function used is based on a spatially subsampled distribution of optical flow magnitude information. The intuitive logic being that in interface applications the target is most often lost during periods of motion, and thus, motion cues should be used as the means of redistributing samples to areas of high target probability.

The optical flow normal components are calculated using an adaptation of the technique outlined in [9] which produces sparse accurate optical flow information. The actual flow vectors produced by this optical flow algorithm suffers from the typical errors experienced by most optical flow techniques; e.g. aperture, blank walls, non-dense measurements, as well as some guess incorrectness. To achieve dense measurements we threshold the flow vectors by an estimate of flow error and subsequently apply a few iterations of numerical methods derived Laplace's equation. We chiefly selected this optical flow technical for its fast performance qualities, which is essential to real-time interface devices. This produces an image map which is sub-sampled with a $\frac{1}{3}$ sample ratio. At this point, a mask can be applied to negate pixels in motion which are not of the color range of interest. From this map, a list of points is produced that satisfies both the optical flow magnitude threshold limit and which satisfy a loose RGB-color definition. This list is then the candidate population for generating sample set particles using importance sampling.

Since some color information has already been included in the importance sampling function at this point, the algorithm in [7] is simplified so that all particles in the sample-with-replacement scheme are selected from either the prior or the importance sampling candidate list. Experimentally it was found that an appropriate division was to select 1 sample from the importance sampling list for every 9 samples selected from the prior. At this point, regardless of how the samples were generated (from the prior or importance sampling) they are all evaluated and weighted using only the color information specified in 4.1.

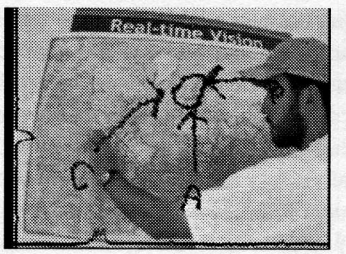

Figure 4: *The visual interface device is shown above as a tool for digitally annotating a wall map.*

5 Experimental Results

The probabilistic tracking algorithm outlined in section 4 was employed in a real-time visual interface systems. The demonstration application is the annotation of a video stream in real-time using a hand-held marker (see figure 4). In figure 4, and in similar figures throughout this paper, the posterior estimate of the x and y target position is plotted on the x and y axis of the image. Since the posterior is only estimated up to an unknown scale factor, there is no magnitude scale displayed for the plots. The sample population size was 1000, as this level guaranteed an accurate posterior representation, as well as allowing for 29 frames/second performance. This trade-off between an accurate representation and effective frame rate is shown in figure 5. As would be

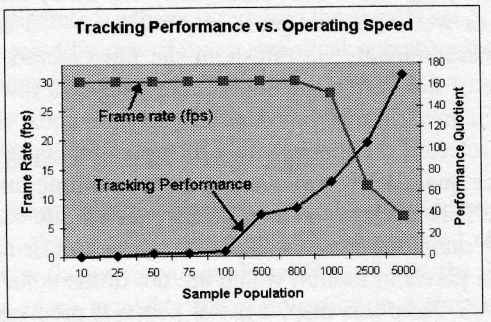

Figure 5: *As the sample population increases, tracking performance increases while the frame rate decreases. A maximum frame rate of 30fps is attainable with sample populations of less than 800.*

expected, as the sample population increases, the effective frame rate decreases and the quality of the tracking results increases. To quantitatively measure tracking performance an MPEG video file was created showing a user tracing a known curve with the hand-held marker. At each time step the deviation between the predicted target location and the closest point on the true curve path was calculated. From this, a Mean Track Deviation (MTD) value can be computed by averaging the total deviation in pixels of the tracker from the plotted course. Due to the stochastic nature of the system, the same video sequence will not produce identical tracking results in different runs of the tracker. Therefore, the tracker was run ten times using the MPEG file as an input at eight different sample population sizes. For each population value, the average MTD value and the variance of the MTD values was calculated. A desirable tracking system will produce both accurate (low MTD) and repeatable (low variance) results. Therefore, a Tracking Performance Quotient (TPQ) was developed to metric the overall quantitative performance. This TPQ value is defined in equation (4) where MTD_{pop} is the set of MTD values calculated for a given population, *pop*.

$$TPQ_{pop} = \frac{1}{Mean(MTD_{pop})Variance(MTD_{pop})} \quad (4)$$

The TPQ was calculated for each set of 10 trial runs at a given sample population. In figure 5 it is shown that the effective frame rate is 29fps for sample populations of less than 1000. At a sample population of 1000 the mean MTD value was 7.60 pixels/frame and the variance of the MTD values was 0.0019 pixels. These represent acceptable performance values for basic video annotation tasks and as a result the recommended population size in our system is 1000 particles. A high frame rate is vital in this application as it constrains the maximum pixels/frame velocity of the target, and thus affects the temporal models Gaussian variance. Compared to other stochastic particle trackers [2, 5], a sample population of 1000 is a relatively small number. We believe the ability of the system to accurately track the target in the presence of clutter and occlusion, with a small number of samples, is largely a result of the use of motion cues to focus the search to regions of high probability in state-space. The focusing ability of these motion cues allows the tracker to investigate a smaller sub-set of state-space than would normally be required for proper posterior estimation.

5.1 Video Annotation

Our interface system for video annotation is a simple one. In order to annotate over top of the video stream the user makes the ball visible to the camera. To temporarily halt the marking process (i.e. to indicate pen-up) , the user hides the ball by enclosing his hand around it (see figure 6). When the tracker recognizes that it has no strong lock on the target location, the annotation is stopped until the target is re-acquired. The color of the annotated graphic is determined by the estimated velocity of the target. This interface could easily be extended to allow additional marker properties (annotation size, style, etc.) to be defined by additional target properties (distance from camera, acceleration, etc.). Determining whether a strong lock is present is done by evaluating the variance of the calculated posteriors. If the variance is below the $Tstrong$ threshold, then it is assumed that a strong lock is present. This relates to the notion that a large posterior variance indicates an element of ambiguity. On screen, a strong target lock is denoted by a dark cross (+) over the target location, while a weak lock is denoted by an x over the target location (see figure 6). The system makes no differentiation between the situation in which the user intentionally occludes the ball by covering it and the situation in which the ball is temporarily occluded by another phenomena (e.g. leaves field of view, occlusion by arm). In each scenario, the marking process is halted since a determination

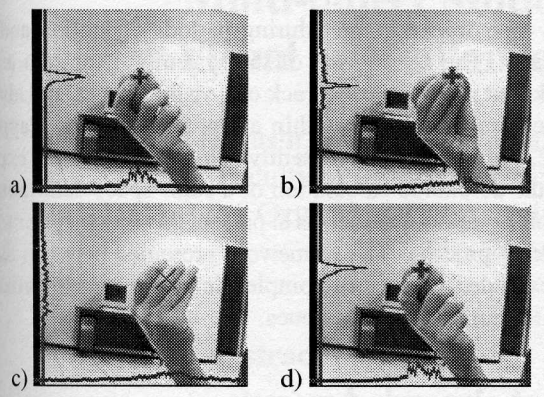

Figure 6: *Intentional Occlusion: This sequence shows the relationship between target visibility and posterior variance.*

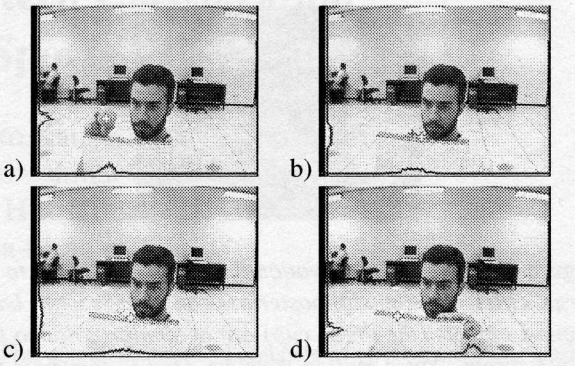

Figure 7: *Recovery from Occlusion: This image sequence shows the ability of our tracker to recover from occlusion. A conventional blob tracker is displayed as a white cross, while our tracker is shown as a dark cross.*

of target location can not be made within a satisfactory level of certainty. The color properties of the observational model were learned prior to the annotation sequence by sampling a distribution of points that were known to be on the ball in the video stream. As stated in section 4.2, the temporal model employed used a random Gaussian diffusion element, coupled with a constant velocity assumption.

5.2 Results

The performance of the video annotation system was promising. At all times when the marker was made visible the tracking system properly locked on to it. The system also proved exceptionally resilient to occlusion, even when the target changed direction during occlusion and reappeared at a location far from its initial point of occlusion. This is due to the use of motion cues in the importance sampling function which removes complete dependence on assumptions of velocity and trajectory.

As expected, during periods of occlusion, the posterior diffused until unambiguous data could be used to strengthen one of the hypotheses. This is clearly shown in the video sequence in figure 7. This sequence shows the target initially visible (i), then hidden behind a book (ii), and then reappearing on the other side of the book after having changed direction (iv). The white cross represents the performance of a conventional blob tracker (discussed in section 4) and the dark cross represents our tracking system feedback. Both trackers perform similarly when the ball is initially visible (i). However, the conventional blob tracker is unable to deal with the occlusion and permanently loses the target after it disappears (iii). Once the ball begins to reappear, motion cues divert a portion of the samples to investigate the source of the motion (iv). When it is observed that the pixels in motion match the observational model well, the corresponding hypothesis begins to strengthen.

As stated in 5.1, the annotation function attempts to determine when the target is visible by the variance of the posterior. The relationship between visibility and posterior variance is illustrated in figure 6. When the ball is clearly visible (i), the posterior is a single mode distribution with a small variance. As diffusion begins (ii) the posterior begins to diffuse to a wider region, although a definitive peak is still observable. Once fully occluded (iii) the posterior has diffused to a large region. This produces a posterior variance significantly larger than that of a visible target. As the ball begins to reappear, the posterior once again converges to the true target location and the variance decreases. The $Tstrong$ threshold value which determines the cut-off between a visible and non-visible target is application dependent and can be determined empirically. For instance, an application in which the target is placed further from the camera will generally produce smaller posterior variances since the target appears smaller in the image.

As stated in section 3, a motivation for using particle filtering is the ability to represent multi-modal posteriors. Multi-modal posteriors typically arise when the observation model is simultaneously attracted to multiple objects in the image. Such a scenario is illustrated in figure 8 in which the green ball and the green cup on the desk both fit the color-specific observational model. This is reflected in both the x and y posteriors which clearly show a distinct peak for each of the objects. Differentiating between objects similar in appearance is dependent on secondary cues (such as the motion information described in section 4.3 and the temporal model). Therefore, despite the observational model being attracted to both of the objects, the true target peak remains dominant since it is being reinforced by the importance sampling motion information.

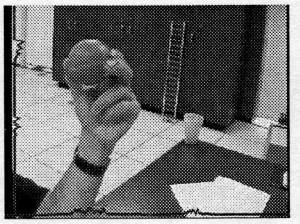

Figure 8: *Since the observational model is attracted to the target color, multi-modal posteriors can emerge when background objects (the green cup) are of similar color to that of the target. Since motion cues are used to focus the target localization, the dominant peak of the posterior remains focused on the true target.*

5.3 Accuracy and Repeatability

As discussed in section 5, the stochastic nature of the system results in multiple instances of the tracking system not returning the exact same target trajectory. However, our system has been shown to produce a low standard deviation of results, and thus, is highly repeatable. To verify this, the average standard deviation of the target state was calculated when 5 trackers were run simultaneously on an MPEG-video clip. The data was organized to present the state variance versus the calculated target velocity. As expected, a larger target velocity produced a greater target variance between the trackers. However, even an exceptionally fast moving target (55+ pixels/frame) produced a low standard deviation (2 pixels) of tracker outputs when using 1000 samples.

The most common source of failure for the tracking system is due to a change in environmental conditions which renders the observational model faulty. This is largely due to changes in lighting which cause the tuned response of the likelihood calculations to be attracted to objects not of the known target color. A possible solution to this problem that is being investigated is to define the observational model color space using the HSV (Hue Saturation Value) index or in terms of normalized red and green channels, where $R = \frac{r}{(r+g+b)}$ and $G = \frac{g}{(r+g+b)}$. This would make the likelihood calculations more robust to shadows and environmental changes.

6 Conclusion

This paper presents a real-time implementation of a visual interface system. The system is demonstrated to perform well as a video annotation tool using a small hand-held marker. Color cues and motion cues were demonstrated to be a natural complement to one another for visual interface devices. This is due to the fact that trackers most commonly lose their tracked target during periods of motion and occlusion. The failure rate of the system is near-zero as the tracker at no point loses track of a visible target, and always re-acquired the target within a few frames of its reappearance. Further work is presently being done which is exploring the automatic calculation of a color-space definition for the observational model, as at present this is done prior to the tracking process. The framework presented here can easily be extended to a more complex interface which utilizes a greater number of image cues.

7 Acknowledgments

The authors express thanks to funding from NSERC (National Science and Engineering Research Council) and MMO (Materials and Manufacturing of Ontario).

References

[1] L. Goncalves, E. Bernardo, E. Ursella, and P. Perona, "Monocular tracking of the human arm in 3d," in *ICCV'95*, pp. 764–770, 1995.

[2] H. Sidenbladh, M. Black, and D. Fleet, "Stochastic tracking of 3d human figures using 2d image motion," in *ECCV'00*, pp. 702–718, 2000.

[3] G. Jang and I. Kweon, "Robust realtime face tracking using adaptive color model," tech. rep., Korea Advanced Institute of Science and Technology, Korea, 2000.

[4] C. Rasmussen, J. Toyamo, and G. Hager, "Tracking objects by color alone," in *Technial Report DCS-TR-1114, Yale University*, 1996.

[5] M. Isard and A. Blake, "Condensation: Conditional density propagation for visual tracking," in *Int. Journal of Computer Vision*, pp. 5–28, 1998.

[6] G. Welch and G. Bishop, "An introduction to the kalman filter," Tech. Rep. TR95041, University of North Carolina, Dept. of Computer Science, Chapel Hill, NC, USA, 2000.

[7] M. Isard and A. Blake, "Icondensation: Unifying low level and high level tracking in a stochastic framework," in *Proc. of ECCV*, pp. 893–908, 1998.

[8] J. Davis and A. Bobick, "The representation and recognition of human movement using temporal templates," in *CVPR 1997*, pp. 928–934, 1997.

[9] T. Camus, "Real time quantized optical flow," in *Journal of Real Time Imaging*, pp. 71–86, 1998.

[10] J. Davis, "Recognizing movement using motion histograms," in *Technial Report 487, MIT Media Lab*, 1999.

[11] O. King, "How does condensation behave with a finite number of samples?," in *ECCV*, pp. 695–709, 2000.

[12] D. MacKay, *Introduction to Monte Carlo Methods, Neural Computation*. 1997.