Technical Report UW-CSE-2007-04-01

Interactive Video Object Annotation

Dan B Goldman[1,2] Brian Curless[1] David Salesin[1,2] Steven M. Seitz[1]

[1]University of Washington [2]Adobe Systems

Abstract

We present interactive techniques for visually annotating independently moving objects in a video stream. Features in the video are automatically tracked and grouped in an off-line preprocess that enables later interactive manipulation and annotation. Examples of such annotations include speech and thought balloons, video graffiti, hyperlinks, and path arrows. Our system also employs a direct-manipulation interface for random frame access using spatial constraints. This annotation interface can be employed in a variety of applications including surveillance, film and video editing, visual tagging, and authoring rich media such as hyperlinked video.

1 Introduction

Annotation is a powerful tool for adding useful information to images. Graphical annotations are often used to highlight regions or objects of interest, indicate their motion, and supply additional contextual information through text or other symbolic markings. For example, a weather map may include satellite images, numbers indicating temperatures, letters representing high and low pressure regions, and schematic elements indicating the direction of the motion of those regions. Such graphical annotations are easily added to still images using modern commercial software such as Adobe Photoshop, which provides an intuitive drawing paradigm using multiple layers of raster and vector graphics.

However, while there are many practical approaches for static image annotation, graphical annotation of arbitrary *video* material is a more challenging task, since the objects or regions move across the screen over time. One approach is the "telestrator," the device used by American football broadcasters to overlay hand-drawn diagrams over still images of sports video (see Figure 1). The telestrator simply allows a static diagram to be drawn atop the video. Typically, these diagrams are removed from the screen while the video is playing, because their location on the screen is aligned with the field or the players only for the single frame upon which they were originally sketched. Another approach is employed by visual effects software such as Adobe After Effects, in which user-specified regions of the image are tracked — typically using normalized cross-correlation template tracking — and various annotations can subsequently be attached to transform along with the tracked regions. In the hands of a skilled artist, this approach can result in annotations that appear "locked" to objects in the video, but it can require substantial manual labor to select the points to be tracked and to correct tracking errors.

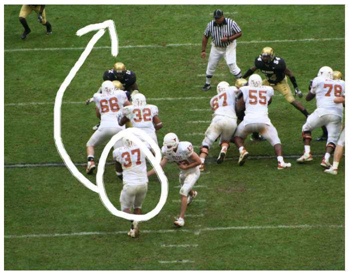


Figure 1 A typical telestrator illustration during a football broadcast. (©Johntex, Creative Commons license [Wikipedia 2006])

In this paper we describe a system that makes it easy to author annotations that transform along with objects or regions in an arbitrary video. Our system first analyzes the video in a fully automatic preprocessing step that tracks the motion of image points across the video and segments those tracks into coherently moving groups. These groups and the motion of the tracked points are then used to drive an interactive annotation interface. We call these annotations "video object annotations" (VOAs) because they are associated with specific objects or regions of the video, unlike telestrator-style annotations that are simply overlaid at a given screen location.

We envision video object annotations being used in any field in which video is produced or used to communicate information. Telestrator-like markup can be useful not only for sports broadcasting but also for medical applications, surveillance video, and instructional video. Film and video professionals can use VOAs to communicate editorial information about footage in post-production, such as objects to be eliminated or augmented with visual effects. VOAs can also be used to modify existing video footage for entertainment purposes with speech and thought balloons, virtual graffiti, "pop-up video" notes, and other arbitrary signage. In this paper, we demonstrate results for several of these applications. Finally, our interface also naturally lends itself to a variety of other applications, such as direct manipulation scrubbing and hyperlinked video authoring.

In this work we describe a pipeline for video processing that greatly enriches the space of interactions that are possible with video, including a fluid and intuitive interface for scrubbing through the time axis of a video and creating graphical annotations. This interaction is enabled in part by a novel feature grouping algorithm.

2 Related work

The telestrator [Wikipedia 2006] is a key point of reference for our system. The telestrator was invented in the late 1960s by physicist Leonard Reiffel for drawing annotations on a TV screen using a light pen. It first became popularly known in 1982 when it was used by color commentator John Madden during instant replays for Super Bowl XVI, and is therefore often colloquially known as a "John Madden-style whiteboard." A similar approach has also been adopted for individual sports instruction using systems like ASTAR [2006] that aid coaches in reviewing videos of athletic performance. However, as previously mentioned, annotations created using a telestrator are typically static, and do not overlay well on moving footage.

In recent years, broadcasts of many professional sporting events have utilized systems supplied by Sportvision [2006] to overlay graphical information on the field of play even while the camera is moving. Sportvision uses a variety of technologies to accomplish these overlays. In most cases, the playing or racing environment and cameras are carefully surveyed, instrumented, and calibrated before play begins. In addition, objects such as hockey pucks and race cars are instrumented with transmitting devices of various kinds so that their locations can be recovered in real time. Finally, chroma keying is sometimes used to matte the players, so that the annotations appear to be painted directly on the field of play. Although this instrumentation and calibration allows graphics to be overlaid in real time during the broadcast, it requires expensive specialized systems for each different class of sporting event, and is not applicable to pre-existing video acquired under unknown conditions.

The visual effects industry has also adopted the concept of the telestrator for facilitating communications between directors and visual effects artists. A product called cineSync [2006] allows individuals in multiple locations to share control of a video file that has been previously transmitted to each location. Both parties can scrub the video and draw annotations on the screen. Because the actual video data is preloaded onto the client computers, no video data is being transmitted during the session. Therefore very little network bandwidth is required: sessions can even be run over 56K modem connections. However, as with a telestrator, the annotations are only associated with still frames.

Tracking has previously been used for video manipulation and authoring animations. For example, Agarwala *et al.* [2004] demonstrated that an interactive keyframe-based contour tracking system could be used for video manipulation and stroke animation authoring. However, their system required considerable user intervention to perform tracking. In contrast, our application does not require pixel-accurate tracking or object segmentation, so we can use more fully-automated techniques that do not produce pixel segmentations. In a related vein, the systems of Li *et al.* [2005] and Wang *et al.* [2005] can be used to segment videos into independently moving objects with considerable accuracy, but do not explicitly recover the transformations of objects over time, and therefore cannot be used to affix annotations. Our system, on the other hand, performs tracking and segmentation as an off-line preprocess, so that new annotations can be created at interactive rates.

Our method utilizes the particle video approach of Sand and Teller [Sand and Teller 2006] to densely track points in the video. Object tracking is a widely researched topic in computer vision, and many other tracking approaches are possible; Yilmaz *et al.* [2006] recently surveyed the state of the art. Particle video is especially well suited to interactive video applications because it provides a dense field of tracked points that can track fairly small objects, and even tracks points in featureless regions.

Our grouping preprocess accomplishes some of the same goals as the object grouping technique of Sivic *et al.* [2006], which tracks features using affine-covariant feature matching and template tracking, followed by a grouping method employing co-occurrence of tracks in motion groups. That method has shown significant success at grouping different views of the same object even through deformations and significant lighting changes. However, after some experimentation we found that it has several drawbacks for our application: First, the field of tracked and grouped points is relatively sparse, especially in featureless areas of the image. Second, affine-covariant feature regions are sometimes quite large, and may therefore overlap multiple moving objects. Finally, the grouping method relies on segmenting a feature similarity matrix using connected components. This process does not scale well to tens of thousands of tracked particles, not only because of the increased memory requirements but also because the connected components approach is not robust to particles that transition from one group to another due to tracking errors. However, unlike Sivic's approach, our grouping method requires that the number of motion groups is given *a priori*.

Our system features a novel interface for scrubbing through video using direct manipulation of video objects. This technique is similar in spirit to the storyboard-based scrubbing approach of Goldman *et al.* [2006], but permits manipulation directly on the video frame, rather than on an auxiliary storyboard image.

Thought and speech balloons have previously been employed in virtual worlds and chat rooms [Morningstar and Farmer 1991; Kurlander et al. 1996], in which the associated regions are known a priori. Kurlander *et al.* specifically address the problem of balloon layout. However, our system allows association of thought and speech balloons with video objects for which the position and movement is not known beforehand.

Our system includes an optimization of annotation location (Section 4.2) that balances proximity to the target with overlap of important features of the image. Related systems have been developed by Thanedar and Höllerer [2004] for pre-recorded video, and by Rosten *et al.* [2005] for augmented reality displays. Our approach in this respect is similar to the work of Rosten *et al.*, but adapts the optimization to the case of pre-recorded video while retaining interactive speeds. Unlike Thanedar and Höllerer, who apply low-level video features to detect regions of low importance, our system uses feature groupings to explicitly detect moving objects in the scene.

We are not the first to propose the notion of hyperlinked video as described in Section 4.5. The earliest reference of this to our knowledge is the Hypersoap project [Dakss et al. 1999]. However, the authoring tool proposed in that work required extensive user annotation of many frames. We believe our system offers a significantly improved authoring environment for this type of rich media.

3 Pre-processing

Our system consists of two off-line preprocessing stages followed by an interactive interface. In the first off-line preprocess, point particles are placed and tracked over time (Section 3.1). Subsequently, we employ a novel grouping mechanism to aggregate particles into consistent moving groups (Section 3.2). The resulting tracked particles and group labels are then used for a variety of applications (Section 4).

3.1 Particle tracking

To track particles, we apply the "particle video" long-range point tracking method [Sand and Teller 2006; Sand 2006], which we briefly recapitulate:

First, we compute optical flow on pairs of consecutive frames, using an energy function that includes a smoothness term modulated by the image gradient magnitude, and an occlusion factor that selects occluding boundaries using the divergence of the flow field and pixel projection differences. Bilateral filtering is applied near flow boundaries to improve boundary sharpness.

Then, particles are propagated from one frame to the next using an optimization process that considers the flow field, image intensity, and color channels, and a weighted smoothness term with respect to nearby particles. At each frame, particles with high post-optimization error are pruned, and new particles are added in gaps between existing particles.

The key advantage of the particle video approach over either template tracking or optical flow alone is that it is both spatially dense and temporally long-range. In contrast, feature tracking is long-range but spatially sparse, and optical flow is dense but temporally short-range. Thus, particle video data is ideal for the purpose of attaching annotations, as we can estimate the motion of any pixel into any frame by finding a nearby particle.

In the sections that follow, we will use the following notation: A particle track i is represented by a 2D position $\mathbf{x}_i(t)$ at each time t during its lifetime $t \in T(i)$.

3.2 Particle grouping

For certain annotation applications, we find it useful to estimate groups of points that move together over time. Our system estimates these groupings using a generative K-affines motion model, in which the motion of each particle is generated by one of K affine motions plus isotropic Gaussian noise:

$$\begin{aligned}\mathbf{x}_i(t+\Delta t) &= A_{L(i)}(t)\mathbf{x}_i(t)+\mathbf{n}\\ \mathbf{n} &\sim N(0,\sigma)\end{aligned}$$

Here $A_k(t)$ represents the affine motion of group k from time t to time $t+\Delta t$, and $\mathbf{n}$ is zero-mean isotropic noise with standard deviation σ. In our system, $\Delta t = 3$ frames. Each particle has a group label $1 \le L(i) \le K$ that is constant over the lifetime of the particle. The labels $L(i)$ are distributed with unknown prior probability $P[L(i)=k]=\pi_k$. We denote group k as $G_k = \{i|L(i)=k\}$.

We optimize for the maximum likelihood model $\Theta = (A_k\forall k, \pi_k\forall k, L(i)\forall i)$ using an EM-style alternating optimization. Given the above generative model, the energy function Q can be simplified to:

$$Q(\Theta)=\sum_i \sum_{t\in T(i)} \left(\frac{d(i,t)}{2\sigma^2} - \log(\pi_{L(i)})\right)$$

where $d(i,t)$ is the residual squared error $||\mathbf{x}_i(t+\Delta t) - A_{L(i)}(t)\mathbf{x}_i(t)||^2$.

To compute the labels $L(i)$, we begin by initializing them to random integers between 1 and K. Then, the following steps are iterated until convergence:

- Affine motions A_k are estimated using the particles G_k.
- Group probabilities π_k are estimated using the numbers of particles $||G_k||$ in each group.
- Labels $L(i)$ are reassigned to the label that minimizes the objective function $Q(\Theta)$ per particle, and the groups G_k are updated.

In the present algorithm we fix $\sigma = 1$ pixel, but this could be included as a variable in the optimization as well.

The output of the algorithm is a segmentation of the particles into K groups. Figure 2 illustrates this grouping on one of our input datasets. Although there are some misclassified particles, the bulk of the particles are properly grouped. Our interactive interface can be used to overcome the minor misclassifications seen here.

4 Applications

Our interactive annotation interface is patterned after typical drawing and painting applications, with the addition of a video time-line. The user is presented with a video window, in which the video can be scrubbed back and forth using either a slider or a novel direct manipulation interface described below. A toolbox provides access to a number of different types of VOAs, which are created and edited using direct manipulation in the video window.

4.1 Video selection

The user creates VOAs simply by painting a stroke s or dragging a rectangle over the region $\mathbf{R}_s$ of the image to which the annotation should be attached. This region is called the annotation's *anchor region*, and the frame on which it is drawn is called the *anchor frame*, denoted t_s. The anchor region defines a set of *anchor tracks* that control the motion of that annotation. For some applications, it suffices to define the anchor tracks as the set of all particles on the anchor frame that lie within the anchor region:

$$\mathbf{A}(s) = \{i|t_s \in T(i), \mathbf{x}_i(t_s) \in \mathbf{R}_s\}$$

However, this simplistic approach to selecting anchor tracks requires the user to scribble over a potentially large anchor region. We can reduce the amount of user effort by employing the particle groupings computed in Section 3.2. Our interface uses the group labels of the particles in the anchor region to infer entire group selections, rather than individual particle selections. To this end, we support two modes of object selection. First, the user can click once to select the group of points of which the closest track is a member. The closest track i' to point $\mathbf{x}_0$ on frame t_0 is located as:

$$i'(\mathbf{x}_0,t_0) = \text{argmin}_{\{i|t_0\in T(i)\}} ||\mathbf{x}_0 - \mathbf{x}_i(t_0)||, \qquad (1)$$

and the selected group is simply $G_\mathbf{x} = L(i'(\mathbf{x},t))$. Second, the user can make a "sloppy" selection that includes points from multiple groups. The resulting selection consists of the groups that are well represented in the anchor region. We score each group by the number $||\mathbf{A}_k(s)||$ of its particles in the anchor region s, then accept any group whose score is a significant fraction T_G of the highest scoring group:

$$\begin{aligned}\mathbf{A}_k(s) &= G_k \cap \mathbf{A}(s) \quad \forall 1\le k\le K\\ S_k(s) &= ||\mathbf{A}_k(s)||/\max_{1\le k\le K} ||\mathbf{A}_k(s)||\\ \mathbf{G}(s) &= \bigcup_{k|S_k(s)>=T_G} G_k\end{aligned}$$

The threshold T_G is a system constant that controls the selection precision. When $T_G = 1$, only the highest-scoring group $\text{argmax}_k||\mathbf{A}_k(s)||$ is selected. As T_G approaches 0, any group with a particle in the selected region will be selected in its entirety. We have found that $T_G = 0.5$ gives very intuitive results in most cases.

Our affine grouping mechanism may group particles together that are spatially discontiguous. However, discontiguous regions are not always appropriate for annotation. To address this we select only the particles that are spatially contiguous to the anchor region. This is achieved using a precomputed Delaunay triangulation of the particles on the anchor frame.

By allowing more than one group to be selected, the user can easily correct the case of over-segmentation, such that connected objects with slightly different motion may have been placed in separate groups. If a user selects groups that move independently, the attached annotations will simply be a "best fit" to both motions.

Using groups of particles confers several advantages over independent particles. As previously mentioned, user interaction is streamlined by employing a single click or a loose selection to indicate a moving object with complex shape and trajectory. Furthermore, the large number of particles in the object groupings can be used to compute more robust motion estimates for rigidly moving objects. We can also display annotations even on frames where the original particles no longer exist due to partial occlusions or deformations. (However, our method is not robust to the case in which an entire group is occluded and later becomes visible again. This is a topic for future work.)

When an object being annotated is partially occluded, we would like to be able to modify its annotation's appearance or location, either to explicitly indicate the occlusion or to move the annotation to an un-occluded region. One indication of occlusion is that the tracked particles in the occluded region are terminated. Although this is a reliable indicator of occlusion, it does not help determine when the same points on the object are disoccluded, since the newly spawned particles in the disoccluded region are not the same as the particles that were terminated when the occlusion occurred. Here again we are aided by the grouping mechanism, since it associates these points on either side of the occlusion as long as there are

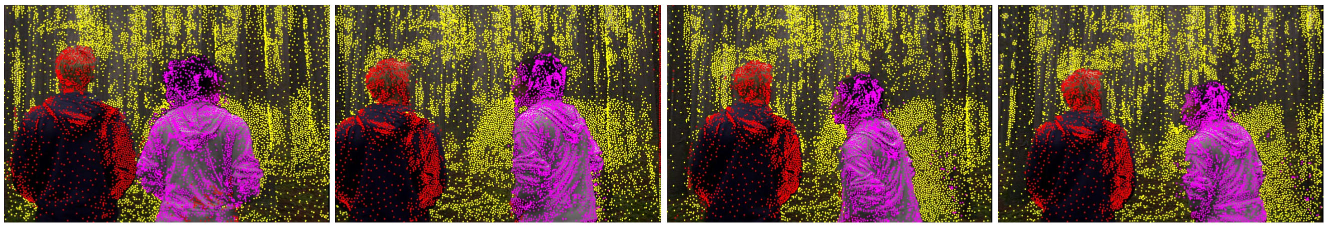


Figure 2 Four frames from a video sequence, with particles colored according to the affine groupings computed in Section 3.2. (video footage ©2005 Jon Goldman)

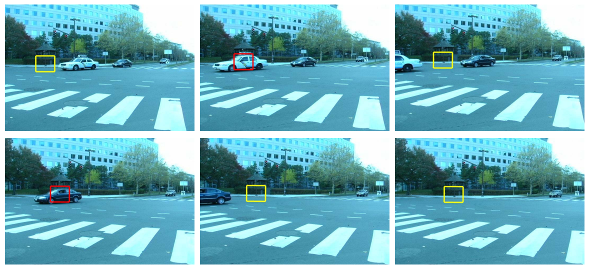


Figure 3 A rectangle created on the first frame remains affixed to the background even when its anchor region is partially or completely occluded. The annotation changes from yellow to black to show that it is occluded.

other particles in the group to bridge the two sets. To determine if a region of the image instantiated at one frame is occluded at some other frame, we simply compute the fraction of particles in the transformed region that belong to the groups present in the initial frame. Figure 3 shows a rectangular annotation changing color as it is occluded and disoccluded.

4.2 Video annotations

Our system supports four types of graphical video object annotations. The types are distinguished both by their shape and by the type of transformations they use to follow the scene. In each case, the transformations of the annotation's anchor tracks are used to determine the appearance and/or transformation of the annotation.

Given the anchor tracks, transformations between the anchor frame and other frames are computed using point correspondences between the features on each frame. Some transformations require a minimum number of correspondences, so if there are too few correspondences on a given frame — for example because the entire group is occluded — the VOA is not shown on that frame.

At present, we have implemented prototype versions of "scribbles," "graffiti," "speech balloons," and "path arrows."

Scribbles. These simple typed or sketched annotations just translate along with the mean translation of anchor tracks. This annotation is ideal for simple communicative tasks, such as local or remote discussions between collaborators in film and video production.

Graffiti. These annotations inherit a perspective deformation from their anchor tracks, as if they are painted on a planar surface such as a wall or ground plane. Given four or more non-collinear point correspondences, a homography is computed using the method described by Hartley and Zisserman [2004]. An example of a graffiti annotation is shown in Figure 4.

When the user completes the drawing of the anchor regions, the transformations of graffiti annotations are not computed for all frames immediately, but are lazily evaluated as the user visits other frames. Further improvements to perceived interaction speed are possible by performing the computations during idle callbacks between user interactions.

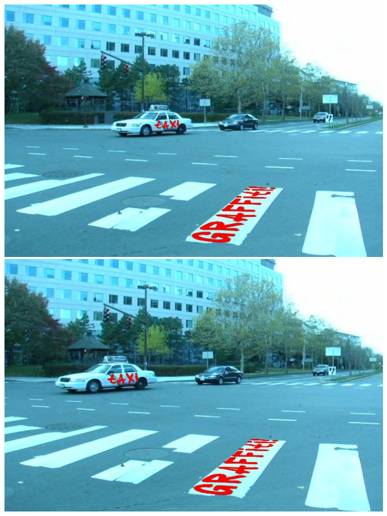


Figure 4 Two "graffiti" annotations attached to a car and a road at two different frames in a sequence.

Speech balloons. Our system implements speech balloons that reside at a fixed location on the screen, with a "tail" that follows the annotated object. The location of the speech balloon is optimized to avoid overlap with foreground objects and other speech balloons, while remaining close to the anchor tracks. Inspired by Comic Chat [Kurlander et al. 1996] and Rosten *et al.* [2005], we optimize an energy function with a distance term E_d, overlap term E_o, and collision term E_c:

$$E = c_d \sum_a E_d(i) + c_o \sum_a E_o(a) + c_c \sum_{a,b} E_c(a,b)$$

where a and b index over the speech balloon annotations.

The distance term E_d simply measures the distance of the speech balloon's mean position $\mathbf{x}_a$ from its anchor tracks:

$$E_d(a) = \sum_t \sum_{i \in \mathbf{A}_a} ||\mathbf{x}_a - \mathbf{x}_i(t)||^2 \tag{2}$$

The overlap term E_o measures the amount by which the annotation overlaps foreground objects:

$$\begin{aligned} E_o(a) &= \sum_{t\in a}\sum_{\mathbf{x}\in a} f(\mathbf{x},t)/V(a) \\ f(\mathbf{x},t) &= \begin{cases} 1, & i'(\mathbf{x},t) \in \mathbf{G}_{fg} \\ 0, & i'(\mathbf{x},t) \notin \mathbf{G}_{fg} \end{cases} \end{aligned}$$

where $V(a)$ is a normalization term representing the spatio-temporal volume of the annotation and i' is the closest track as defined in equation (1). Here we use the notational shorthand $\mathbf{x} \in a$ to refer to points inside the balloon region, and $t \in a$ to refer to the balloon's duration of existence. By default, our system defines background points as those belonging to the largest group

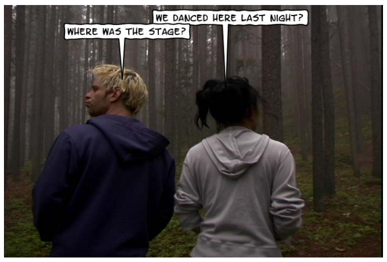


Figure 5 Two speech balloons with screen position optimized to minimize overlap with the actors throughout the shot. (video footage ©2005 Jon Goldman)

$\mathbf{G}_{bg} = \mathbf{G}_{\mathrm{argmax}_k ||\mathbf{G}_k||}$, and all other points belong to the foreground ($\mathbf{G}_{fg} = \{i | i \notin \mathbf{G}_{bg}\}$), but the user can easily indicate different foreground and background groups using the selection mechanism described in Section 4.1.

Finally, the collision term E_c measures the amount by which multiple annotations overlap, and it is computed analogously to E_o.

Many terms in the energy function can be factored and/or approximated in order to accelerate computation. For example, notice that Equation 2 can be rearranged as:

$$E_d(a) = N||\mathbf{x}_a||^2 - 2\mathbf{x}_a^T \sum_{t,i} \mathbf{x}_i(t) + \sum_{t,i} ||\mathbf{x}_i(t)||^2$$

The third term is a constant over the optimization, and can therefore be omitted from the energy function, and the sum in the second term can be computed once before the optimization process. The computation of E_o can be accelerated by precomputing summed area tables for the expression in the inner loop, and by approximating the shape of a thought balloon using its bounding rectangle. E_c can also be computed in constant time for the case of rectangular regions. Using all these accelerations, we are able to compute a robust global minimum for E in a few seconds using BFGS with several random initializations for the two thought balloons in Figure 5. (See the companion video [Goldman et al. 2007] for a real-time demonstration.) In our implementation, $c_o = c_c = 10000$, and $c_d = 1$.

We also experimented with animated speech balloons that only translate or translate and scale with their anchor tracks, and also using a global optimization over all frames with a constraint to enforce smooth motion. However, we found that speech balloons moving about the screen were difficult to read, even when moving quite slowly. Our present implementation is therefore designed to maximize legibility at the risk of some overlap and awkward tail crossings. In the rare case in which annotated objects change location dramatically on the screen, e.g., by crossing over each other from left to right, this implementation may result in undesirable layouts with crossed tails or balloons that do overlap foreground objects. However, we note that it is extremely atypical in modern cinematography for characters to swap screen locations while talking. In reviewing several full length and short films we found less than a dozen such shots. In every case the dialog was not fully overlapping, so that speech balloons could appear over different frame ranges in order to avoid occlusions and crossed tails.

Path arrows. These annotations highlight a particular moving object, displaying an arrow indicating its motion onto a plane in the scene. To compute the arrow path we transform the motion of the centroid of the anchor tracks into the coordinate system of the background group in each frame. This path is used to draw an arrow that transforms along with the background.

By computing a rough matte for the moving objects, we can also matte the arrow so that it appears to lie behind the moving subject. The rough matte we use for these visual effects is obtained using the group label of the closest particle for each pixel. (Higher quality mattes can be obtained using a variety of existing methods, at additional computational cost [Rother et al. 2004; Wang and Cohen 2005; Wang et al. 2005; Levin et al. 2006].) Our result can be seen in Figure 6. We believe this type of annotation could be used by surveillance analysts, or to enhance telestrator-style markup of sporting events.

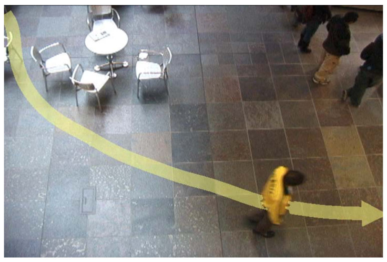


Figure 6 An arrow highlighting the motion of a walking person.

4.3 Scrubbing via direct manipulation

Since we have already densely tracked the video, we can scrub to a different frame of the video by directly clicking and dragging on any moving object. This UI is implemented as follows: When the user clicks at location $\mathbf{x}_0$ while on frame t_0, the closest track i' is computed as in equation (1), and the offset between the mouse position and the track location is retained for future use: $\mathbf{d} = \mathbf{x}_0 - \mathbf{x}_{i'}(t_0)$. Then, as the user drags the mouse to position $\mathbf{x}_1$, the video is scrubbed to the frame t' in which the offset mouse position $\mathbf{x}_1 + \mathbf{d}$ is closest to the track position on that frame:

$$t' = \mathrm{argmin}_{\{t \in T(i')\}} ||\mathbf{x}_1 + \mathbf{d} - \mathbf{x}'_i(t)||$$

Figures 7 and 8a)-c) illustrate this behavior.

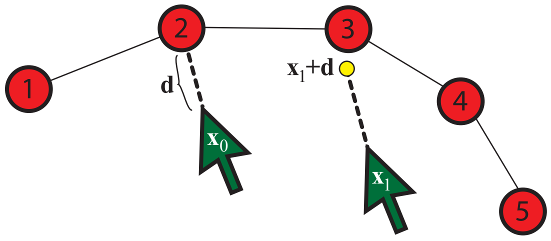
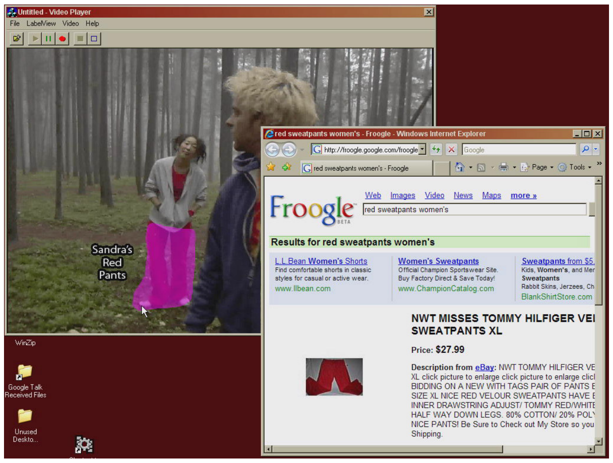


Figure 7 When the mouse is depressed at position $\mathbf{x}_0$ on frame 2, the feature track shown in red is selected as the closest track. When the mouse moves to position $\mathbf{x}_1$, the feature at frame 3 of that track is closer to the offset mouse position $\mathbf{x}_1 + \mathbf{d}$, so the video is scrubbed to frame 3.

4.4 Constraint-based video control

By extending the scrubbing technique described above to multiple particle paths, we also implement a form of constraint-based video control. The user sets multiple point constraints on different parts of the video image, and the video is advanced or rewound to the frame that minimizes the sum of squared distances from the particles to their constraint positions. Here, c indexes over constraint locations $\mathbf{x}_c$, offsets $\mathbf{d}_c$, and constrained particles i'_c:

$$t' = \mathrm{argmin}_{\{t \in T(i')\}} \sum_{c \in C} ||\mathbf{x}_c + \mathbf{d}_c - \mathbf{x}_{i'_c}(t)||^2$$


Untitled - Video Player
File LabelView Video Help
Sandra's Red Pants
red sweatpants women's - Froogle - Windows Internet Explorer
http://froogle.google.com/froogle
Google
red sweatpants women's - Froogle
Page
Tools
Froogle BETA
Web Images Video News Maps more »
red sweatpants women's
WinZip
Google Talk Received Files
Unused Deskto...

Results for red sweatpants women's

L.L.Bean Women's Shorts
Find comfortable shorts in classic styles for casual or active wear.
www.llbean.com

Women's Sweatpants
Official Champion Sportswear Site. Buy Factory Direct & Save Today!
www.ChampionCatalog.com

Sweatpants from $5
Kids, Women's, and Men
Sweatpants
Rabbit Skins, Jerzees, Ch
BlankShirtStore.com

NWT MISSES TOMMY HILFIGER VE
SWEATPANTS XL

Price: $27.99

Description from eBay: NWT TOMMY HILFIGER VE
XL click picture to enlarge click picture to enlarge clic
BIDDING ON A NEW WITH TAGS PAIR OF PANTS E
SIZE XL NICE RED VELOUR SWEATPANTS HAVE E
INNER DRAWSTRING ADJUST/ TOMMY RED/WHITE
HALF WAY DOWN LEGS. 80% COTTON/ 20% POL
NICE PANTS! Be Sure to Check out My Store so you
Shipping.

Figure 9 A video with highlighted hyperlinks to web pages. (video footage ©2005 Jon Goldman)

In our mouse-based interface, the user can set a number of fixed-location constraints and one dynamic constraint, controlled by the mouse. However, multiple dynamic constraints could be applied using a multitouch input device. Figures 8d) and 8e) illustrate facial animation using multiple constraints.

4.5 Multimedia authoring

Our system can also be used to author alternative media representations of the original video, such as hyperlinked video [Dakss et al. 1999].

A prototype hyper-video player using our system as a front end for annotation is shown in Figure 9, and can also be seen in the companion video [Goldman et al. 2007]. When viewing the video on an appropriate device, the user can obtain additional information about objects annotated in this way, for example, obtaining price information for clothing or other depicted items, or additional references for historically or scientifically interesting objects. As a hyperlink, this additional information does not obscure the video content under normal viewing conditions, but rather allows the viewer to actively choose to obtain further information when desired. The hyperlinked regions in this 30-second segment of video were annotated using our interface in about 5 minutes of user time.

5 Discussion

We have presented a system for interactively associating graphical annotations to independently moving video objects. Our contributions include the application of an automated preprocess for video interaction, a novel grouping algorithm for tracked points, a fluid interface for creating graphical annotations that transform along with associated video objects, and a novel interaction technique for scrubbing through video.

A primary benefit of our approach over existing methods for annotating video is that we perform tracking as an off-line preprocess. This means that the user does not need to be in the loop for selecting regions to be tracked and correcting the results. Instead, we ask the user only to perform higher level tasks such as selecting objects to be annotated and the types of annotations to be used. Furthermore, our grouping preprocess allows for rapid and intuitive selection of coherently moving regions.

Although traditional image mosaicking techniques can be used to scrub video by clicking on points in the background [Irani and Anandan 1998], our approach permits manipulations that can't be achieved using previous mosaicking approaches, such as those shown in Figure 8.

Our current scrubbing mechanism has some drawbacks. First, it relies on individual particles that may not survive through an entire shot due to occlusions, deformations, or varying illumination. Therefore it may not always be possible to drag an object through its entire range of motion using a single click and drag. We believe this can be resolved by taking advantage of our object groupings: Nearby particles in the same affine motion group can be used to extend the dragging motion beyond the endpoints of an individual particle's path.

Another drawback is that when a video features objects with repetitive or small screen-space motions — like a subject moving back and forth along a single path, or moving directly toward the camera — it may be hard or impossible to reach a desired frame using this mechanism. It is possible that heuristics could be applied to infer the user's intent in such cases. However, we submit our interface not as a replacement for traditional scrollbars and jog/shuttle widgets, but rather as a supplementary mode of manipulation.

One important limitation of our system is the length of time required to preprocess the video. In our current implementation, the preprocess takes about 10 minutes per frame of 720×480 input video, which is prohibitive for some of the potential applications described here. Although most of the preprocess is heavily parallelizable, and moderate accelerations can be attained by tuning the code, novel algorithms will be necessary for applications requiring "instant replay."

Another limitation is that our grouping mechanism does not enforce spatial coherence, imposing some additional burden of effort on the user in cases where the motion alone is not sufficient to separate multiple objects. We would like to explore the use of spatial constraints and image properties in our grouping algorithm, to combine the benefits of existing pixel segmentation algorithms with our novel motion segmentation approach.

6 Future work

In spite of some limitations, we believe our approach to interactive video annotation may have a number of applications in a variety of domains. We envision the following scenarios as a sampling of future extensions to our work:

In film and video production, interactive annotations can be used by editors and directors to communicate about objects and individuals by making markings directly on their footage. A director can simply circle the objects she wants removed or emphasized, and the annotation is viewable on all frames of the video instantly, remaining aligned with the object of interest. Our technique is easily adapted to remote review settings like that supported by cineSync, since the data associated with each annotation is quite small and can be transmitted in real time over a network connection. Furthermore, we can utilize our interface to author schematic storyboards for pre-production applications [Goldman et al. 2006].

In sports broadcasting, video object annotations could be used to supplant existing telestrator technologies, with the advantage that the annotations can remain on the screen while the video is playing. Furthermore, the lines drawn to illustrate player motion could be generated automatically like a storyboard. In contrast to the Sportvision technologies, no complex and expensive instrumentation of the field of play and the competitors is necessary, so our approach is potentially applicable to historical sports video, lower-budget high school and collegiate sporting events, or even individual sports instruction.

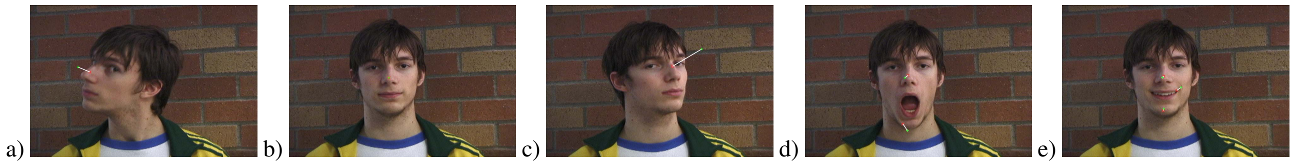
a) b) c) d) e)


Figure 8 Our scrubbing interface is used to interactively manipulate a video of a moving head by a-c) dragging it to the left and right. Additional constraints are applied to d) open the mouth, or e) keep the mouth closed and smile.

Additional applications may be possible by integrating our system with the approach of Sivic *et al.* [2006] to associate the same object in multiple shots, or in multiple videos taken from different cameras. For example, we can imagine extending the Photo Tourism concept [Snavely et al. 2006] to Video Tourism. A viewer could nonlinearly travel through multiple videos that captured the same environment or event, like Michael Naimark's "Moviemaps" of Aspen and Banff. In addition, text or graphical annotations could be propagated from photos to videos, or from one video to another.

Video object annotations can be used to annotate the objects and processes in an assembly instruction video. If the end user also has a camera, the user's current stage of assembly could be detected, and the instructional video synchronized to the appropriate step. A similar technique could be used to synchronize a video of a walking tour to the user's current location.

In another application, an analyst reviewing surveillance video could easily mark individuals of interest for further review using our system. The video may be automatically re-centered or cropped and zoomed on these individuals for easier review. If the scene has been filmed with multiple surveillance cameras, it may also be possible to click on an individual and propagate the annotation into the other videos. Finally, the path of a single individual over time might be followed across multiple cameras automatically.

In conclusion, we believe interactive video object annotations can become an important tool for augmenting video as an informational and interactive medium, and we hope that this research has advanced us several steps closer to that goal.

Acknowledgments

The authors would like to thank Peter Sand and Michael Black for the use of their source code, Sameer Agarwal and Josef Sivic for helpful discussions about motion grouping, Nick Irving at UW Intercollegiate Athletics for sample video footage, and Chris Gonterman, Harshdeep Singh, Samreen Dhillon, Kevin Chiu, and Mira Dontcheva for additional technical assistance. Special thanks to Jon Goldman for the use of footage from his short film *Kind of a Blur*, which can be purchased online at Apple's iTunes Store. Funding for this research was provided by NSF grant EIA-0321235, the University of Washington Animation Research Labs, the Washington Research Foundation, Adobe, and Microsoft.

References

AGARWALA, A., HERTZMANN, A., SALESIN, D. H., AND SEITZ, S. M. 2004. Keyframe-based tracking for rotoscoping and animation. *ACM Trans. Graph. 23*, 3, 584–591.

ASTAR LEARNING SYSTEMS, 2006. Video analysis for coaches, instructors, and more. http://www.astarls.com. [Online; accessed 3-September-2006].

CINESYNC, 2006. share your vision. http://www.cinesync.com. [Online; accessed 29-August-2006].

DAKSS, J., AGAMANOLIS, S., CHALOM, E., AND V. MICHAEL BOVE, J. 1999. Hyperlinked video. In *Proc. SPIE*, vol. 3528, 2–10.

GOLDMAN, D. B., CURLESS, B., SEITZ, S. M., AND SALESIN, D. 2006. Schematic storyboarding for video visualization and editing. *ACM Transactions on Graphics (Proc. SIGGRAPH) 25*, 3 (July), 862–871.

GOLDMAN, D. B., CURLESS, B., SEITZ, S. M., AND SALESIN, D., 2007. Interactive video object annotation (website). http://grail.cs.washington.edu/projects/ivoa/tr07/. [Online; accessed 26-April-2007].

HARTLEY, R. I., AND ZISSERMAN, A. 2004. *Multiple View Geometry in Computer Vision*, second ed. Cambridge University Press, ISBN: 0521540518.

IRANI, M., AND ANANDAN, P. 1998. Video indexing based on mosaic representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence 86*, 5 (May), 905–921.

KURLANDER, D., SKELLY, T., AND SALESIN, D. 1996. Comic chat. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 225–236.

LEVIN, A., LISCHINSKI, D., AND WEISS, Y. 2006. A closed form solution to natural image matting. In *CVPR*, 61–68.

LI, Y., SUN, J., AND SHUM, H.-Y. 2005. Video object cut and paste. *ACM Trans. Graph. 24*, 3, 595–600.

MORNINGSTAR, C., AND FARMER, R. F. 1991. The lessons of Lucasfilm's Habitat. In *Cyberspace: First Steps*, M. Benedikt, Ed. MIT Press, Cambridge, MA, 273–301.

ROSTEN, E., REITMAYR, G., AND DRUMMOND, T. 2005. Real-time video annotations for augmented reality. In *Proc. International Symposium on Visual Computing*.

ROTHER, C., KOLMOGOROV, V., AND BLAKE, A. 2004. "GrabCut" – interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (Proc. SIGGRAPH) 23*, 3, 309–314.

SAND, P., AND TELLER, S. 2006. Particle video: Long-range motion estimation using point trajectories. In *Proc. CVPR '06*, vol. 2, 2195–2202.

SAND, P. 2006. *Long-Range Video Motion Estimation using Point Trajectories*. PhD thesis, Massachusetts Institute of Technology.

SIVIC, J., SCHAFFALITZKY, F., AND ZISSERMAN, A. 2006. Object level grouping for video shots. *International Journal of Computer Vision 67*, 2, 189–210.

SNAVELY, N., SEITZ, S. M., AND SZELISKI, R. 2006. Photo tourism: Exploring photo collections in 3D. *ACM Transactions on Graphics (Proc. SIGGRAPH) 25*, 3, 835–846.

SPORTVISION, 2006. Changing The Game. http://www.sportvision.com. [Online; accessed 29-August-2006].

THANEDAR, V., AND HÖLLERER, T. 2004. Semi-automated placement of annotations in videos. Tech. Rep. 2004-11, UC, Santa Barbara.

WANG, J., AND COHEN, M. 2005. An iterative optimization approach for unified image segmentation and matting. In *ICCV*, vol. 2, 936–943.

WANG, J., BHAT, P., COLBURN, R. A., AGRAWALA, M., AND COHEN, M. F. 2005. Interactive video cutout. *ACM Trans. Graph. 24*, 3, 585–594.

WIKIPEDIA, 2006. Telestrator — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/w/index.php?title=Telestrator&oldid=64269499. [Online; accessed 28-August-2006].

YILMAZ, A., JAVED, O., AND SHAH, M. 2006. Object tracking: A survey. *ACM Computing Surveys 38*, 4 (December), 13.