Video Retrieval based on User-Specified Appearance and Application to Animation Synthesis

Makoto Okabe[1] Yuta Kawate[2] Ken Anjyo[3] Rikio Onai[4]

[1,2,4]The University of Electro-Communications, Tokyo, Japan
[3]OLM Digital, Inc. / JST CREST
[1]JST PRESTO
[1]m.o@acm.org [2]kawate@onailab.com [3]anjyo@olm.co.jp [4]onai@cs.uec.ac.jp

Abstract. In our research group, we investigate techniques for retrieving videos based on user-specified appearances. In this paper, we introduce two of our research activities.
First, we present a user interface for quickly and easily retrieving scenes of a desired appearance from videos. Given an input image, our system allows the user to sketch a transformation of an object inside the image, and then retrieves scenes showing this object in the user-specified transformed pose. Our method employs two steps to retrieve the target scenes. We first apply a standard image-retrieval technique based on feature matching, and find scenes in which the same object appears in a similar pose. Then we find the target scene by automatically forwarding or rewinding the video, starting from the frame selected in the previous step. When the user-specified transformation is matched, we stop forwarding or rewinding, and thus the target scene is retrieved. We demonstrate that our method successfully retrieves scenes of a racing car, a running horse, and a flying airplane with user-specified poses and motions.
Secondly, we present a method for synthesizing fluid animation from a single image, using a fluid video database. The user inputs a target painting or photograph of a fluid scene. Employing the database of fluid video examples, the core algorithm of our technique then automatically retrieves and assigns appropriate fluid videos for each part of the target image. The procedure can thus be used to handle various paintings and photographs of rivers, waterfalls, fire, and smoke, and the resulting animations demonstrate that it is more powerful and efficient than our prior work.

1 Video Retrieval by User-Specified Transformation

Because the number of accessible videos is growing larger by the day (especially on the Internet), many people are interested in ways to quickly and easily find

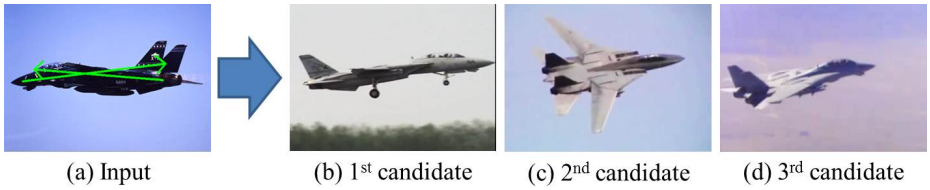

Fig. 1. (a) The current video frame and the user-drawn green arrows specifying the transformation: the tip of the fighter aircraft moves toward the right and the tail comes from the left. (b-d) The scenes retrieved from the video collection by our method, with the aircraft flying from left to right.

certain scenes in these videos. Therefore, video retrieval has become an active research area for computer vision and multimedia specialists.

Most video search engines, such as YouTube [1] and gettyimages [2], currently support only text input in queries, and video searches are based on verbal information manually assigned to each video (e.g., the title of a video or tags). These engines do not understand queries pertaining to a desired pose or motion of a video object. If we input a text query such as "I want to watch a fighter aircraft flying from left to right," none of the existing video search engines can retrieve scenes such as those shown in Fig. 1-b, c, or d. Furthermore, even if we were to develop a smart video search engine capable of understanding such text queries, it would be difficult or tedious for human users to precisely describe a desired pose or motion with mere words.

To address this problem, we are interested in an appearance-based user interface that enables the interactive exploration of video collections. Google Video and its extensions propose image-based image or video retrieval [1, 2] by representing each video frame as a relatively low-dimensional vector, using bag-of-features. Photo tourism and related methods allow the user to interactively explore photo and video collections by walking through a three-dimensional (3D) scene reconstructed from the collections [3–6]. These are based on precise 3D reconstruction of the scene and camera positions via the incremental structure-from-motion method. However, the technique is typically applied only to stationary objects such as buildings, is difficult to apply to moving, deformable objects, and is computationally expensive. Direct object manipulation also allows interactive navigation of scenes in a single video. [7–11]. The user navigates a video by dragging a video object and interactively editing its posture. Because these techniques are based on two-dimensional (2D) video processing rather than 3D reconstruction, their computational cost is relatively low. However, we want to perform this type of video object navigation not only in a single video, but also in video collections.

[1] http://www.youtube.com
[2] http://www.gettyimages.com

We propose an appearance-based interface that allows the user to quickly and easily specify the desired pose and motion of a video object. The user first specifies the input image by simply pausing a video or preparing some other image. Then the user specifies a transformation of an object inside the image by drawing arrows, using our sketching interface (Fig. 1). After several seconds, our system displays the candidate scenes retrieved from the video database. In these scenes, the object from the input image is transformed (i.e., rotated, scaled, or translated) in the 3D world according to the user's specifications. Nevertheless, all user input into our system is 2D, which allows the user to design a query intuitively. Our algorithm also relies only on 2D image- and video-processing technologies (i.e., does not reconstruct any 3D information), which keeps the computational cost low. We demonstrate that our method can be successfully applied to different types of video objects, including a racing car, a running horse, and a flying airplane. We also carry out a subjective evaluation of the usability of our system.

1.1 Our Approach

Let's assume that we are now watching the scene of Fig. 2-a, where a red racing car is running from left to right. At the same time, we feel like watching another scene like Fig. 2-b, where a similar car is running toward the front. To find such a desired scene, we usually manipulate the video player many times, e.g., by pushing the forward and rewind buttons or moving the play bar: if we cannot find the desired scene in the currently watching video, we have to go to a video search engine like YouTube or gettyimages to further search for it. However, these are tedious tasks.

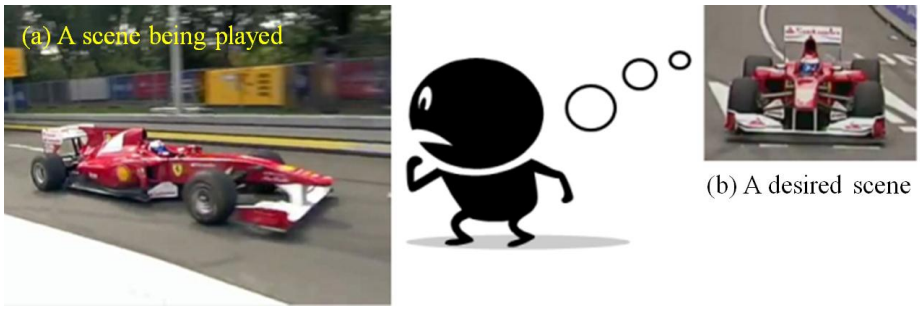

Fig. 2. Our motivation.

To support this user to find the desired scene, we propose a user interaction for video retrieval. Using our sketching interface, the user specifies the transformation of the object, i.e., the red car in this case, and then the system automatically retrieves the candidates of the user-desired scene. Fig. 3-a shows the exmaple. The user draws the two green arrows, which specify where the front

and back spoilers should come in the desired scene. Fig. 3-b is the result that our system actually retrieved from the database: we overlap the same arrows over the image, which show each spoiler has come at the corresponding tip of each arrow.

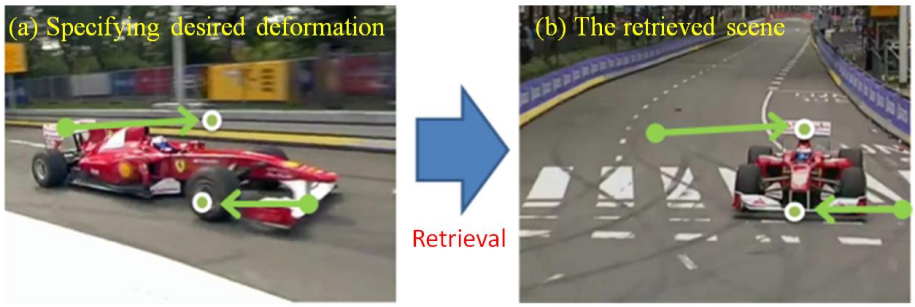

Fig. 3. The proposed user interaction.

1.2 Algorithm

It is difficult for any existing image or video retrieval technique to directly retrieve the desired scene (Fig. 2-b) using the currently watching scene (Fig. 2-a) as the query. For example, using the simple method of feature matching, we extract SIFT (Scale-Invariant Feature Transform) features [12] in both the currently watching frame and every frame of the videos in the database, and compute the matching between them. Fig. 4 shows the results. In Fig. 4-left, the top and bottom frames are similar looking scenes: the red car is left-side-right but the posture of the car is almost the same. As a result, we successfully find many consistent matches between them as the many yellow lines, and the computer can retrieve the bottom frame as the result. On the other hand, there is no consistent match in the right case: the red car is the same, but since the bottom car is the rotated version of the top car. SIFT describes only 2D image feature and does not capture this 3D rotation.

However, it is interesting to note that the bottom two frames in Fig. 4 exist in the same video sequence as shown in Fig. 5. This fact lets the computer find the user-desired frame (Fig. 4-bottom right): the computer can find the frame of Fig. 5-d, and then find the desired scene by rewinding the video from the frame until Fig. 5-a is found. During the rewinding process, we track the front and back spoilers, since they are specified by the user as tails of the arrows. We stop to rewind the video, when each tracker comes near to the tip of the corresponding arrow. Then, we reach the desired scene.

In conclusion, our algorithm employs two steps (Fig. 6): we first find a frame that looks similar to the query frame, using SIFT matching, and then we automatically forward or rewind the video from that frame to find the user-desired

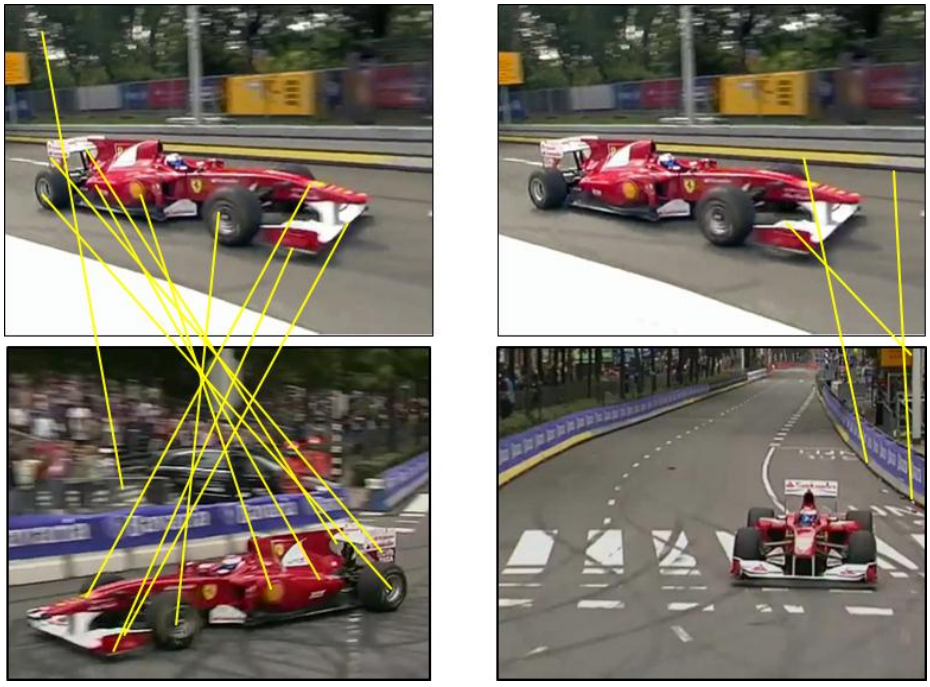

Fig. 4. Video retrieval using SIFT matching. Left: We find many consistent matches. Right: We cannot find any consistent match.

scene. SIFT matching is efficiently performed using a kd-tree algorithm. To efficiently find the desired scene by forwarding or rewinding the video, we use the particle video algorithm [13] for motion tracking. As shown in the right part of Fig. 6, we distribute the particles throughout the image space and track their underlying motion. The red particles represent those newly added in the given frame, while the blue particles are continuing from the previous frame. In the pre-processing stage of the database construction, all the particles and their trajectories are computed and saved. In the forwarding or rewinding process, the system selects the particles around the starting points of the user-drawn arrows in the frames found in the SIFT matching step. During the forwarding

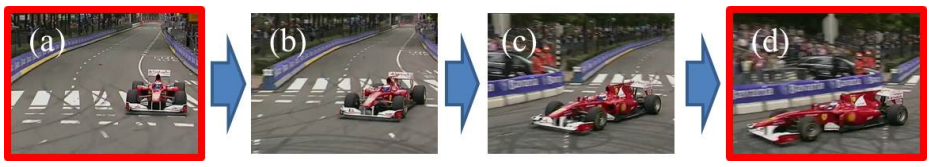

Fig. 5. (a) The red car is coming toward the front (Fig. 4-bottom-right), and then (d) turning to the left (Fig. 4-bottom-left).

or rewinding process, the system always checks whether or not their trajectories pass through the tips of the arrows at the same time. If so, the frame corresponding to that time is output as the retrieved scene.

Fig. 6. A summary of our algorithm.

1.3 Results and Discussion

We tested our method on three types of video collections obtained from YouTube: videos of racing cars, running horses, and flying airplanes. All videos had a resolution of 320×240. After downloading videos from each category, we created a video database a priori. We computed the SIFT features for every frame of each video. We also performed motion tracking using the particle video algorithm [13], and saved all particle trajectories in the database. All experiments were performed using a desktop PC with an Intel i7-860 2.8 GHz processor and 4.0 GB of memory.

Fig. 1 and Fig. 7 show the set of input images, the arrows drawn by the user, and the results of our video retrieval. Detailed statistics from our experiments are listed in Table 1. Given the user input, our system expends an average of about 4 seconds on the retrieval process.

Video Type	# of Frames	Time for Retrieval (seconds)
Car	3718	4.43
Horse	4450	3.10
Aircraft	3741	4.04

Table 1. The statistics of our experiments. The number of video frames and the average time spent for the video retrieval are shown.

The first and second rows of Fig. 7 show the retrieval of the racing car videos. In the first row, the user draws two arrows in an attempt to find scenes of a car moving from right to left. One arrow specifies that the back spoiler moves from the right, and the other specifies that the tip of the car moves toward the left.

The three columns of results show the best, second best, and third best retrieved scenes. All of them show the car traveling toward the left, which matches the user's specification. In the second row, the user again draws two arrows on the front and back spoilers, to find scenes of the scaled-down car moving forward. In these results, the car has virtually the same pose in each result, but the scale varies from one result to the next. This is caused by inaccuracies in the motion tracking of the particle video algorithm; each particle often slides over the video object, especially parts moving at high speeds.

Fig. 1 shows the results for the flying airplane, and indicates one limitation of our technique. The user draws two arrows in an attempt to find scenes of a an airplane flying from left to right. In all of the retrieved results, the user's specification has been achieved. However, in the second best candidate, the airplane is also rotated about its medial axis. Even if the user does not desire such rotation, it is often difficult to eliminate it using our system.

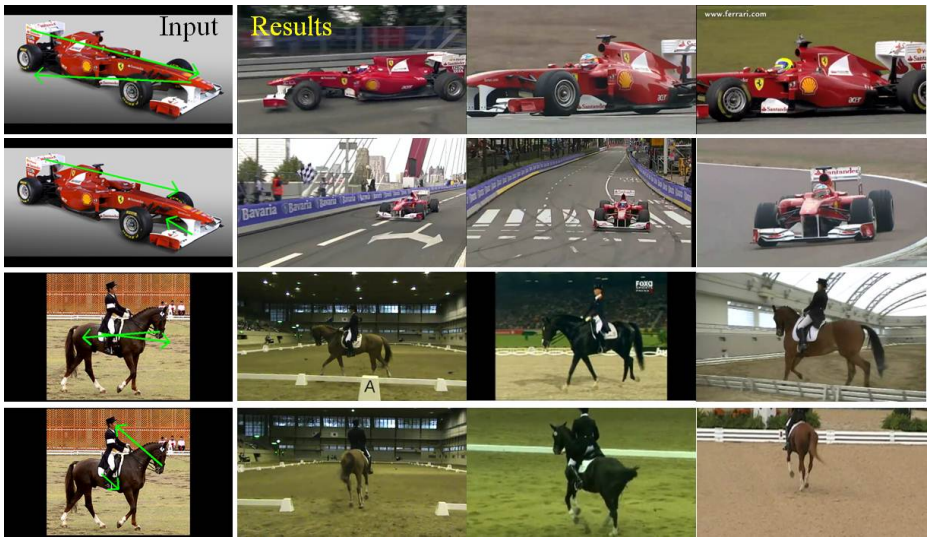

Fig. 7. Retrieval results.

User Study We carried out a user study to investigate the usability of our system. The subjects were nine students from the computer science department, accustomed to watching videos on the Web, but unfamiliar with our systems. We asked each of them to retrieve a desired scene using our system. We showed the car and horse images to each subject (the input images of Fig. 7), and asked the subject to envision a scene and describe it in words. Then the subject drew the arrows on the image, and the system retrieved the three best candidate scenes. The subject watched all of the retrieved scenes, and then compared them to

what he/she had in mind. All subjects tried the car scene, and six of them also tried the horse scene. We asked each subject to repeat the retrieval five times, and counted the number of cases in which the subject found the desired scene.

The score for the car scene was 3.00 ± 0.70 and the score for the horse scene was 3.00 ± 1.41. We only showed the subjects the input images of the car and the horse, and none of them had any prior knowledge of what kinds of scenes were included in the video database. As a result, subjects often specified impossible transformations of the object, and could not find the desired scene. For example, one subject wanted the car moving from the bottom to the top of the image space, but there was no aerial view of the car in our video database. The subjects frequently commented on the difficulty of finding a number of types of scenes (e.g., a scene in which the car is running backward, showing its rear). It is actually difficult to find such a scene via a single interaction, because the rear of the car is invisible in the input image of Fig. 7, and it is impossible to specify an arrow on the rear. However, it is interesting to note that our method does allow the user to find a scene of the rear of the car by repeating the sketching and retrieval operations, rotating the car step-by-step.

2 Creating a Fluid Animation from a Single Image based on Video Retrieval

Creating quality fluid animations is time consuming for computer graphics designers. There are two major methods. In one, physics-based simulation, it is difficult to set the appropriate physical parameters to achieve the desired appearance. The other, making a composite from a video recording of fluids, requires the time-consuming tasks of finding an appropriate video, cutting and pasting the segments accurately, and adjusting the appearance of the composite. We are interested in animating a picture of a fluid to quickly and easily achieve the desired appearance.

A previous study successfully designed fluid animations from pictures, but it was limited to synthesizing relatively calm fluid motions such as water surfaces [14]; our study focuses on synthesizing more dynamic motions such as water splashes. Another method allows users to specify a video example and then transfers its fluid features to the target image [15]; however, only a single video example is used, which limits the available variation in fluid features.

To address these problems, we developed a data-driven method for creating a fluid animation from a picture (Fig. 8). The user inputs a target image (Fig. 8-b) with a few hints about fluid motion (i.e., flow direction and speed) such as sketches of flow direction, shown as orange arrows. The user also specifies an alpha matte that extracts the fluid region of interest (Fig. 8-c). We constructed a video database that includes hundreds of video examples of fluids (Fig. 8-a) and helps the user synthesize better quality animation with less effort than in previous methods (Fig. 8-d). The technical detail is described in our paper [16].

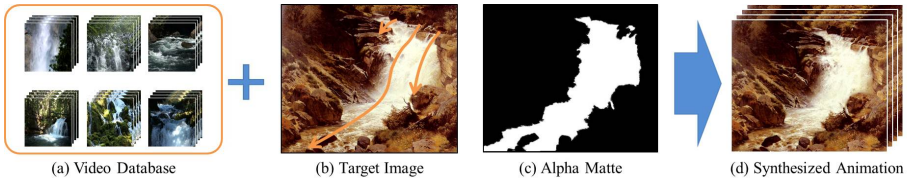

Fig. 8. Creation of a fluid animation from a picture.

2.1 Our Method

Our system consists of three components: 1) construction of a video database of fluids (Fig. 9-a), where each video example is cut into small pieces; 2) a best-match search for an appropriate video example piece and assignment of this to part of the target image; and 3) synthesis of the final animation through seamless integration of the assigned pieces and adjustment of the overall appearance.

The offline process of database construction begins with gathering original video examples of fluids (Fig. 9-b). To increase the number of examples, we cut each video example into small pieces (Fig. 9-c). For each video example piece, we then compute the average image by averaging the frames (Fig. 9-e) to obtain representative information about its appearance. We also calculate the differences between the average image and frames (Fig. 9-f) that have no significant color properties but that capture high-frequency fluid features. Finally, from the averaged images in the database, we construct a bag-of-features codebook and describe each average image using a histogram of visual words (Fig. 9-d).

Hence, we cut a target image (Fig. 9-g) into small pieces using the same process used for database construction (Fig. 9-i). Next, we compute the histogram of visual words for each piece (Fig. 9-h), perform a best-match search between histograms of visual words (see Figs. 9-d and h), and assign video example pieces that are similar to the target-image piece. When a user-specified motion field is given, it is used as a constraint for solving the assignment problem. Based on the assignment results, differences (Fig. 9-f) are copied onto the corresponding target-image piece (Fig. 9-j). Finally, all assigned differences are integrated seamlessly, and the animation is synthesized by adjusting the appearance (Fig. 9-k).

2.2 Results

We constructed an independent database for water, fire, and smoke scenes. This involved gathering 151, 96, and 89 video examples for the water, fire, and smoke databases, respectively, from which we obtained 246, 227, and 195 thousand video example pieces. We synthesized the fluid animation for each target image, as shown in the supplementary video. We designed an alpha matte and specified an orientation map for each target image. We specified a speed map only for the waterfall painting, the fire painting, and the smoke painting.

A side-by-side comparison shows that our method was better at reproducing the fluid features of original video examples than the previous method. In partic-

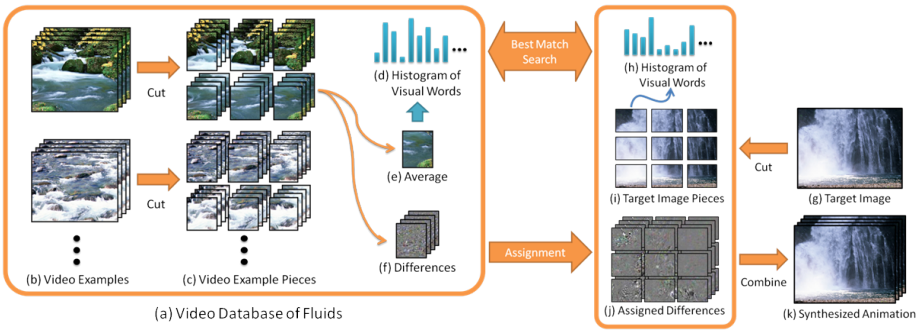

Fig. 9. System overview.

ular, the dynamic water splashes of the video examle were not well reproduced in the previous method; it looks as if irrelevant synthesized noises flow along a static motion field. On the other hand, our method successfully reproduces the fluid features of assigned video example pieces. Our method also made it easier for users to create a fluid animation. For example, using the previous method, the waterfall painting had to be divided into the waterfall and river parts; in contrast, our method could process the whole image at once.

We also performed a user study in which 16 participants ranked the visual quality of each animation. All of the water scenes were given high scores, but fire and smoke scenes scored lower. The smoke in the train scene was difficult to animate because the motion of smoke became chaotic due to a failure in video assignment, and the lower smoke in the scene had visible artifacts of noise caused by video compression that were hidden in the original videos.

3 Conclusion

We have developed the appearance-based user interfaces for video retrieval and the application of the video retrieval technique to animation synthesis. In our approaches, we start with a single image as the input, and then introduce the additional user's suggestions. In the first study, it was the user-specified deformation. In the second study, it was the user-specified motion field of the fluid flow. In both studies, we relied on the sketch-based user interface, i.e., drawing the arrows. We adopted it because we thought it was intuitive for the user. We have demonstrated that the combination of content-based image and video retrieval technique with a few additional suggestions enabled us to propose a novel interaction for video retrieval and animation synthesis.

References

1. Sivic, J., Zisserman, A.: Video google: A text retrieval approach to object matching in videos. In: ICCV. (2003) 1470–1477

2. Datta, R., Joshi, D., Li, J., Wang, J.Z.: Image retrieval: Ideas, influences, and trends of the new age. ACM Comput. Surv. **40**(2) (2008) 5:1–5:60
3. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: exploring photo collections in 3d. In: ACM SIGGRAPH 2006 Papers. (2006) 835–846
4. Agarwal, S., Snavely, N., Simon, I., Seitz, S., Szeliski, R.: Building rome in a day. In: ICCV 2009. (2009) 72 –79
5. Frahm, J.M., Pollefeys, M., Lazebnik, S., Zach, C., Gallup, D., Clipp, B., Raguram, R., Wu, C., Johnson, T.: Fast robust large-scale mapping from video and internet photo collections. ISPRS Journal of Photogrammetry and Remote Sensing **65**(6) (2010) 538–549
6. Tompkin, J., Kim, K., Kautz, J., Theobalt, C.: Videoscapes: Exploring sparse, unstructured video collections. In: ACM Transactions on Graphics (Proc. of SIGGRAPH). (2012)
7. Kimber, D., Dunnigan, T., Girgensohn, A., Shipman, F., Turner, T., Yang, T.: Trailblazing: Video playback control by direct object manipulation. In: IEEE International Conference on Multimedia and Expo. (2007) 1015 –1018
8. Girgensohn, A., Kimber, D., Vaughan, J., Yang, T., Shipman, F., Turner, T., Rieffel, E., Wilcox, L., Chen, F., Dunnigan, T.: Dots: support for effective video surveillance. In: Proc. of ACM Multimedia. (2007) 423–432
9. Dragicevic, P., Ramos, G., Bibliowitcz, J., Nowrouzezahrai, D., Balakrishnan, R., Singh, K.: Video browsing by direct manipulation. In: Proc. of CHI '08. (2008) 237–246
10. Goldman, D.B., Gonterman, C., Curless, B., Salesin, D., Seitz, S.M.: Video object annotation, navigation, and composition. In: Proc. UIST 2008. (2008) 3–12
11. Karrer, T., Weiss, M., Lee, E., Borchers, J.: Dragon: a direct manipulation interface for frame-accurate in-scene video navigation. In: Proc. of CHI '08. (2008) 247–250
12. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vision **60** (2004) 91–110
13. Sand, P., Teller, S.: Particle video: Long-range motion estimation using point trajectories. In: Proc. of CVPR '06. (2006) 2195–2202
14. Chuang, Y.Y., Goldman, D.B., Zheng, K.C., Curless, B., Salesin, D.H., Szeliski, R.: Animating pictures with stochastic motion textures. In: Proc. SIGGRAPH 2005. (2005) 853–860
15. Okabe, M., Anjyo, K., Igarashi, T., Seidel, H.P.: Animating pictures of fluid using video examples. Computer Graphics Forum (Proc. EUROGRAPHICS) **28**(2) (2009) 677–686
16. Okabe, M., Anjyo, K., Onai, R.: Creating fluid animation from a single image using video database. Comput. Graph. Forum **30**(7) (2011) 1973–1982