SIViP (2015) 9:1717–1726
DOI 10.1007/s11760-014-0631-x

ORIGINAL PAPER

HandVR: a hand-gesture-based interface to a video retrieval system

Serkan Genç · Muhammet Baştan · Uğur Güdükbay · Volkan Atalay · Özgür Ulusoy

Received: 8 November 2013 / Revised: 24 December 2013 / Accepted: 25 February 2014 / Published online: 1 April 2014

Abstract Using one's hands in human–computer interaction increases both the effectiveness of computer usage and the speed of interaction. One way of accomplishing this goal is to utilize computer vision techniques to develop hand-gesture-based interfaces. A video database system is one application where a hand-gesture-based interface is useful, because it provides a way to specify certain queries more easily. We present a hand-gesture-based interface for a video database system to specify motion and spatiotemporal object queries. We use a regular, low-cost camera to monitor the movements and configurations of the user's hands and translate them to video queries. We conducted a user study to compare our gesture-based interface with a mouse-based interface on various types of video queries. The users evaluated the two interfaces in terms of different usability parameters, including the ease of learning, ease of use, ease of remembering (memory), naturalness, comfortable use, satisfaction, and enjoyment. The user study showed that querying video databases is a promising application area for hand-gesture-based interfaces, especially for queries involving motion and spatiotemporal relations.

S. Genç
Department of Computer Technology and Information Systems, Bilkent University, 06800 Bilkent, Ankara, Turkey

M. Baştan
Department of Computer Engineering, Turgut Özal University, 06010 Keçiören, Ankara, Turkey

U. Güdükbay (✉) · Ö. Ulusoy
Department of Computer Engineering, Bilkent University, 06800 Bilkent, Ankara, Turkey
e-mail: gudukbay@cs.bilkent.edu.tr

V. Atalay
Department of Computer Engineering, Middle East Technical University, 06531 İnönü Bulvarı, Ankara, Turkey

Keywords Hand-gesture · Query interface · Usability · Motion query · Video retrieval · Multimedia databases

1 Introduction

Convenience of the interaction device is one of the main criteria for effective human–computer interaction (HCI). There is no one interface suitable for all applications; the one that is the most effective and comfortable for the users is the most appropriate. For example, the human hand provides a very promising interface for certain applications. In fact, hand interfaces are supersets of mouse-based interfaces because the capabilities of a mouse can be easily simulated by a hand. Hand-based systems, however, have poor pixel accuracy, due generally to the low-resolution cameras used to monitor the hands and the image processing involved. Fortunately, not all applications require high pixel accuracies, such as the proposed video query interface.

At first, electromechanical gloves that track the user's palm and fingers were used for hand interaction, but this method was uncomfortable, expensive, and complex. An alternative is a vision-based system, in which one or more cameras monitor the user's hands, and the images acquired from the cameras are processed to determine the command the user performs. This system does not require any wearable or attached electromechanical equipment, that is, it is a passive interface. Furthermore, it only needs an inexpensive camera for image acquisition, and it provides the user with a completely immersive interaction experience. However, there are still some problems with such vision-based systems. First, robust real-time vision systems have not yet been developed; many vision applications work under some constraints, e.g., static background, markers for the users. Second, further research is needed to determine appropriate

vision-based hand interface applications and evaluate their performance. This paper contributes to the second problem by investigating whether querying a video database system is an appropriate application for a vision-based hand interface.

Video retrieval systems provide fast access to large video collections, and they are becoming important due to the growing amount of video data [1]. *BilVideo-7* [2] is one such system; it supports text, color, texture, shape, location, motion, and spatiotemporal object queries on videos. First, videos are decomposed into their building blocks: shots, keyframes, still, and moving regions. Then, features are extracted and stored in a database, which can be queried via a visual query interface.

The primary goal of the visual query interface is to provide users with a powerful and easy-to-use interface to formulate various queries on a video database and retrieve the most relevant videos or video segments. Some queries are easy to formulate using keywords (query by keyword), for example, "Retrieve video segments that contain an *airplane*." In this case, the user enters the keyword(s) using the keyboard, presses the *enter* key to execute the query, and waits for the results. If the query interface has speech recognition capability, the keywords can also be entered with speech. Some queries are easier to formulate using examples (query by example) or sketches, such as retrieving video segments having similar color, texture, motion, etc., to the video segment or sketch that is provided by the user.

We designed our hand-gesture-based query interface to support four different query types: (1) spatial queries, (2) motion trajectory queries, (3) motion trajectory with temporal relation queries, and (4) camera motion queries [2]. In the query interface, the hands represent stationary or moving objects (query types 1, 2, and 3), or a camera in different types of motion (query type 4). Therefore, the hands need to be continuously monitored to keep track of their configurations; this is done by a camera connected to a PC running a low-level visual processing module which processes the video from the camera to track the hands.

Our system employs one webcam to monitor the user's hand movements. To facilitate and speed up the hand monitoring process (robustly detect, recognize, and track the hands in real time), the user wears uniformly colored gloves on her hands. Time-of-flight or projective depth cameras (MS Kinect) [3] can also be used for this purpose within the proposed framework instead of wearing different-colored gloves, but since our focus is to investigate the applicability of a hand-gesture-based interface for query formulation in a video database, our colored glove-based system works well for our purposes.

We conducted a user study to evaluate and compare our gesture-based interface with a mouse-based interface. The users performed predefined test queries for each query type on both interfaces and filled out a questionnaire. We evaluated the results to judge the suitability of a hand-gesture-based interface for different types of queries in a video database. The results showed that the gesture-based interface is better for camera motion and motion trajectory with temporal relation queries.

2 Related work

Although the idea of using hands in HCI is promising, it is not effective in all applications. Appropriate applications for hand interfaces should be determined by considering users' assessments of usability criteria. Some applications focused on low-level visual processing in hand interface systems [4,5]. However, there exists limited literature on the usability of hand-based systems. To the best of our knowledge, there are no hand-based interfaces for video retrieval systems in the literature; thus, we aim to fill this gap.

Barehand interaction is desirable in hand-based vision systems; however, it is challenging to achieve robust real-time visual processing performance to monitor the hands. Kolsch and Turk [6] developed a vision-based hand gesture interface, HandVu, to detect and track a hand for wearable mobile applications. It tracks hands well but requires high processing power. Segen and Kumar [7] used barehand interaction for their GestureVR system, but it requires a uniform background. Stauffer and Grimson [8] proposed a method that updates the background model adaptively. This method gives good results for gradual background changes, but it fails in indoor environments where sudden changes may occur frequently. Since our focus is on hand-gesture interface, rather than the low-level visual processing for hand monitoring, we decided to use colored gloves to reduce the problems due to low-level visual processing.

Wang and Popovic [9] developed a system that can reconstruct a 3D hand pose from a single image of the hand wearing a specially designed multi-colored glove. Although there are many sensor-based glove hardware systems that can track and reconstruct a hand robustly [10], Wang and Popovic's system is more comfortable. Since our system does not need 3D model of hand, we sticked to uniformly colored gloves instead of a specially colored or sensor-based gloves.

Numagucci et al. [11] proposed a puppet system to retrieve movie clips that are similar to the motions of a puppet from a motion capture database. They proposed an algorithm to match puppet motions from a movie clip database. Our system enables a user to describe the objects, their motions, and spatiotemporal relations between them by hands. This differs from the study of Numagucci's in two ways. Our system does not use any external objects such as a puppet to define the motion, and the hands describe the objects and their relations, not the motion of one person.

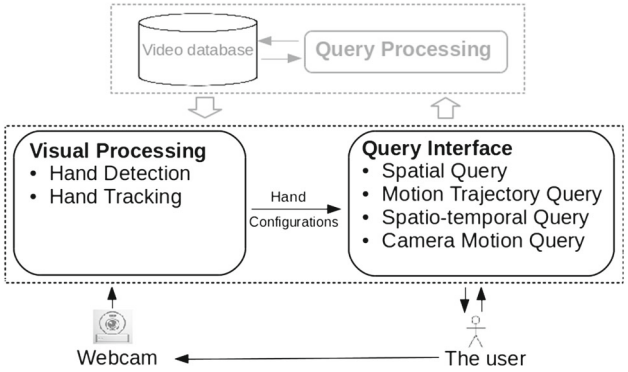

Fig. 1 Software architecture

3 Setup and software architecture

In our hand-gesture-based interface, the user formulates queries using her hands (one or both hands, depending on the query) without wearing electromechanical devices. The user's hands are monitored by a camera. The height of the camera determines the available working space of the user. The user wears two different-colored gloves on her hands. The gloves can be in any color provided that they are uniformly colored, and the background does not contain the colors of the gloves. The system is trained once for a pair of gloves (colors), as described in Sect. 5.1. We also assume that severe lighting changes do not occur after training the system for a pair of gloves (colors), in which case, the system needs to be retrained. These constraints are imposed to enable real-time visual processing, which is important for an interactive system.

Our system has two main modules: (1) *visual processing* module acquires images from the camera and processes them to keep track of the hands' configurations, which are fed to the (2) *query interface* (Fig. 1). The query interface interacts with the user (get inputs, display results, hand properties) and forms the queries according to the query language of the video database system.

4 The hand-gesture-based query interface

The hand-gesture-based query interface forms the video queries based on the spatiotemporal hand configurations, computed by the visual processing module, and displays the query results (Fig. 3). It forms the queries according to the query-by-example (or query-by-sketch) paradigm, that is, the hands represent video objects (or the camera, in camera motion queries), and queries are formed using the raw spatiotemporal hand configurations, rather than converting them to high-level semantics. For example, suppose the user wants to search for video segments where an object moves from left to right (motion trajectory query). To formulate the query, the user moves her hand (the object) from left to right (see the description of the motion trajectory query in Sect. 4.2 for more details); in the meantime, the visual processing module tracks the hand and collects a list of spatiotemporal object locations *(x, y, size, orientation, time)*, which are then used by the query interface directly to form the trajectory query (query-by-sketch). Then, the query processing module can execute and return the query results to the query interface.

4.1 Gesture vocabulary

We define the basic hand gestures used to formulate certain types of video retrieval queries. All queries are a combination of the following gestures and/or postures.

Open-hand posture represents an object whose properties can be modified, as in Fig. 2a. These properties include hand position on the x–y plane (object location), hand orientation (object's angle around the z-axis), and hand size (object size, or zoom level in camera motion queries). While the hand has this posture, changing its properties modifies the corresponding object's properties. Moving

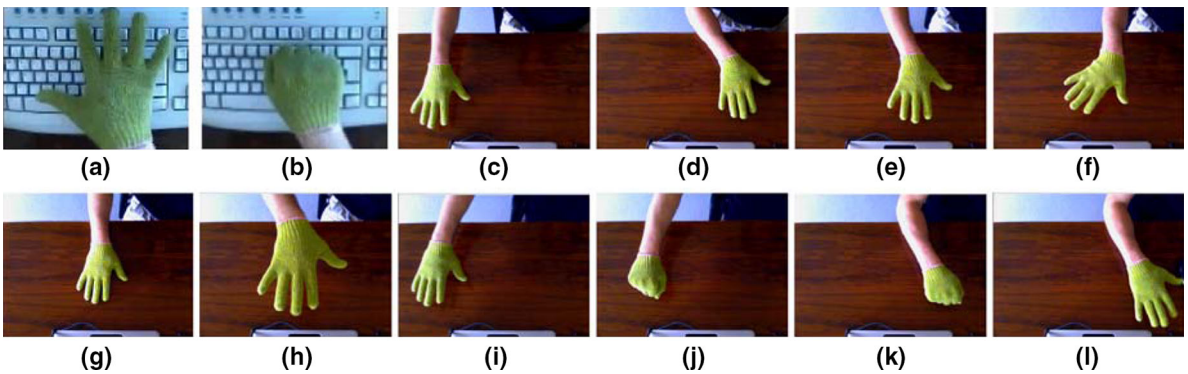

Fig. 2 Gesture/posture vocabulary: **a** open-hand posture; **b** closed-hand posture; **c** and **d** move; **e** and **f** rotate; **g** and **h** zoom or scale; **i–l** trajectory specification

an open-hand results in translation (see Fig. 2c, d), rotating an open hand rotates the object (see Fig. 2e, f), and moving an open hand close to the camera increases the object size or zooms the camera out (see Fig. 2g, h).
Closed-hand posture notifies the hand interface system to add the object with the previous open-hand properties into the query (see Fig. 2b).
Trajectory gesture starts with an open-hand posture to start the trajectory specification. Then, the user closes her hand and specifies the trajectory of the object. To finish the trajectory, she switches from the closed-hand posture to the open-hand posture (see Fig. 2i–l). This gesture is recognized with a finite state machine (FSM).

4.2 Query formulation

Our query interface covers four types of queries: *spatial, motion trajectory, motion trajectory with spatiotemporal relations* and *camera motion*. For each type of query, a separate query interface is implemented.

4.2.1 Spatial Queries

The spatial query is used to search for an object's position, size, and orientation (x, y, s, θ), as well as the spatial relation between objects. One hand represents one object. The open-hand posture describes the properties of the objects in the scene (see Fig. 3a). After an open-hand posture, the user closes her hand to add the object itself into the scene (see Fig. 3b). For example, Fig. 3 shows the user's hands representing a *man* and an *airplane* objects, with their properties (x, y, s, θ). The yellow line shows the object's orientation (θ), and the line's length represents the object's size (s). The blue hand in closed-hand posture adds the *airplane* with the given properties in Fig. 3c into the scene. In this example, the following query is sketched by our hand interface: "A *man* at the right side of the scene is looking at an *airplane* moving diagonally upward to the right."

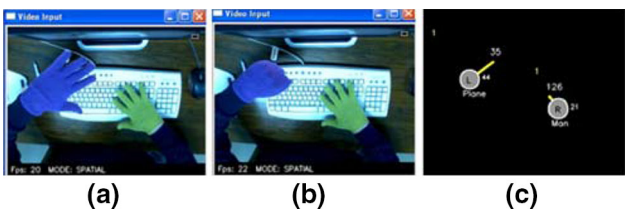

Fig. 3 Spatial query formulation. **a** Initial hand configuration; **b** "Add object" hand posture for *airplane*; **c** hand properties in the scene

4.2.2 Motion trajectory queries.

The motion trajectory query is used to search for objects by their trajectories (paths). To specify a trajectory, the user first opens her hand (open-hand posture) to mark the initial configuration of the object. Then, she closes and moves her hand (closed-hand posture) to specify the trajectory. Assume that the user wants to retrieve the video segments for the query "Find the videos where *John* approaches a stationary *Car* from the left." Figure 4a–d show how the user adds the trajectory of *John* with her left hand. The left hand (blue glove) represents *John*, and the right hand (green glove) represents the *Car*. The user places *John* in his initial position in 4a with the open-hand posture. She marks the beginning of the path with the closed-hand posture for *John* in 4b, then draws his path in 4c, and finishes the path with the open-hand posture again in 4d. In the end, the trajectory is obtained by the visual processing module as a list of spatiotemporal points as shown in 4e.

The start and end of the trajectory is detected with the help of a FSM, shown in Fig. 5. The machine has two states; S_1 is the trajectory start/ end state, and S_2 is the trajectory drawing state. Initially, the user moves her hand in open-hand posture to the starting position of the path. To begin the trajectory, she performs the closed-hand posture for half a second (10 consecutive frames at 20 fps). This causes a transition from S_1 to S_2, marking the starting point of the trajectory. In S_2, while the hand is in the closed-hand posture, all hand positions (x, y) are added to the trajectory. To end the trajectory, the user holds the open-hand posture for another half a second (10 frames).

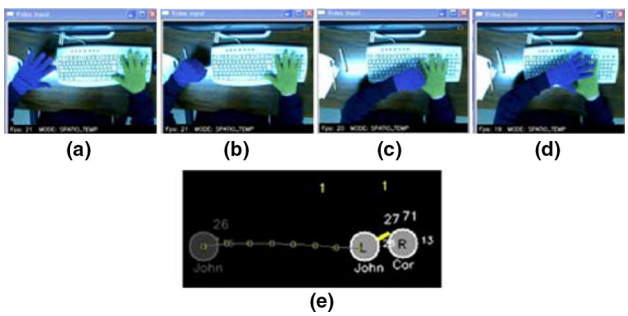

Fig. 4 Motion trajectory query formulation. **a**–**d** Sequence of hand configurations; **e** specifying an object with its path

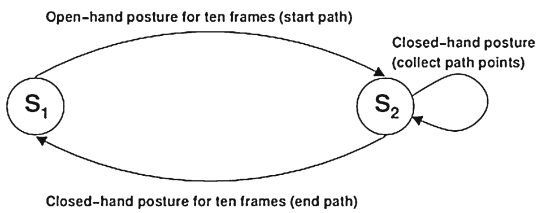

Fig. 5 FSM to recognize a motion trajectory action

4.2.3 Spatiotemporal motion trajectory queries

This type of query involves trajectories with spatiotemporal relations between the objects, e.g., two cars colliding in the middle of the scene. To formulate this query, the user places her hands on the sides and then moves them to each other and collides them in the middle, as shown in Fig. 6. In such queries, specifying the trajectory and spatiotemporal relations with the hands is easier, compared to a mouse-based interface.

4.2.4 Camera motion queries

Camera motion query is used to search for video segments having specified camera motions, such as tracking and zooming. A hand represents the camera, and its position (x, y) is specified by moving the hand on the x–y plane. Zoom in/out (s) is realized by moving the hand in the z-direction (Fig. 7), and hand orientation determines the camera view direction (θ). As the user moves her hand, the visual processing module accumulates a list of camera motion parameters (x, y, s, θ) by tracking the hand.

A sample camera motion query could be "Find all video segments where a camera is moving from left to right while it is panning from left to right and zooming in." Figure 7 shows how to formulate this query using one hand. The hand (or the camera) is placed at the start point, as in Fig. 7a; then, while moving her hand from left to right, the user moves her hand down and changes its orientation from left-facing to right-facing, as in Fig. 7b, c. As can be seen from this example, the user can easily convert a detailed textual query into hand movements. Moreover, the user can easily specify the motion parameters (e.g., location, speed, zoom level, orientation) of the camera at the same time. Such a query is very difficult to formulate using a mouse-based interface.

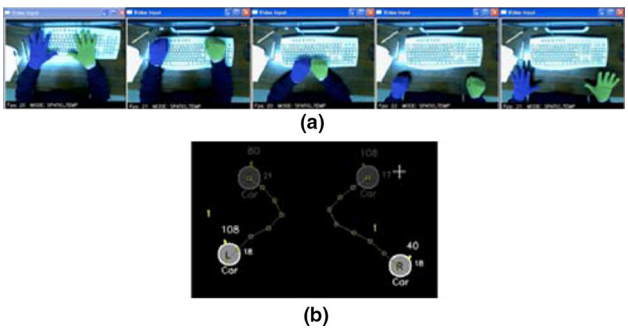

Fig. 6 Spatiotemporal motion trajectory query formulation example: two cars colliding in the middle of the scene. **a** Sequence of hand configurations for collision; **b** defining a collision query

Fig. 7 Camera motion query formulation. **a**–**c** Hand poses for camera motion query; **d** computed spatiotemporal camera configurations

5 Visual processing for hand tracking

The visual processing module keeps track of the hands' configurations (position, size, orientation) and feeds them to the query interface. It uses a fast thresholding based color segmentation to detect the hands wearing uniformly colored gloves and a decision tree classifier with shape features extracted from the segmented hand regions to recognize the hand postures. We preferred simple and fast algorithms for real-time performance, since real-time processing is crucial in HCI.

5.1 Hand detection

Hand detection is achieved via color segmentation of the hands wearing uniformly colored gloves (blue, green). The input frames are segmented into three regions: left hand, right hand, and background (Fig. 8). We use a fast thresholding based segmentation in HSL (hue, saturation, lightness) color space. For each color channel, two threshold values (low, high) are required; hence, for segmenting, one colored glove 6 thresholds are needed $([H_{\text{low}}, H_{\text{high}}], [S_{\text{low}}, S_{\text{high}}], [L_{\text{low}}, L_{\text{high}}])$. In total, 12 threshold values must be determined for two different colors. This needs to be done only once for a pair of colored gloves. The user selects a rectangular hand region for each color/hand on the image to see the histograms of the H, S, and L channels. The histograms are smoothed to reduce the noise, and the thresholds are determined automatically based on the histograms.

The acquired RGB image is converted to HSL color space and processed pixelwise to determine pixels that belong to the hands. After detecting the pixels possibly belonging to the two hands/gloves, noise pixels are eliminated using a

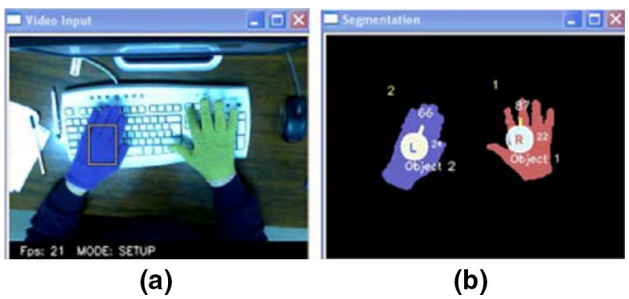

Fig. 8 Color-based hand segmentation. **a** Input image; **b** corresponding segmentation result

3×3 median filter. The remaining pixels are grouped into regions by 8-connected component analysis. Finally, for each color class, the region with the largest area is taken as the hand region; the remaining regions, if any, are discarded. Figure 8b shows an input image and its segmentation. The hand configuration (position, size, orientation) is computed based on this segmentation. The hand position is the center of mass of the hand blob; the hand size is the area of the hand blob; and the hand orientation is in the direction of the major axis of the ellipse obtained by least-squares fitting to the boundary of the hand region.

Open- and closed-hand gestures are determined by using a FSM that models the states (hand shapes) and transitions. The states and transitions are manually constructed to detect the open- and closed-hand gestures (cf. Fig. 5). In Fig. 8b, L and R indicate the left and right hands, respectively; the lines from the L and the R show the orientations of the hands, and the line length is proportional to the hand size.

5.2 Hand posture recognition

Our query interface uses two hand postures (Sect. 4.1) that must be recognized. This is achieved with a decision tree classifier using shape descriptors computed from the segmented hand regions. Among many existing classifiers, we selected the decision tree classifier, for its simplicity and efficiency [12].

There are many shape representation techniques for shape recognition [13]. Considering our problem of recognizing open- and closed-hand postures, which is relatively easy, we selected the following shape descriptors to represent the hand shapes: *compactness*, *axis ratio*, *convexity*, and *rectangularity*. This method is widely used to represent hand shapes and is easy to implement, fast, and robust. Moreover, the descriptors are invariant to translation, rotation, and scaling [12], which is also crucial in hand shape recognition in our case.

The classifier is trained for the two hand postures on various appearances of the hand postures as shown in Fig. 9, using the above four shape descriptors as features. The classifier achieves 98 % accuracy on the task of recognizing our open-/closed-hand postures, and above 90 % on the 8 postures shown in Fig. 10.

6 Experiments

We conducted a user study to assess the usability of the proposed hand-based interface and a mouse-based interface in terms of performance and attitude criteria [14]. The attitude criteria are *usefulness*, *learning*, *memory*, *naturalness*, *comfort*, *satisfaction*, and *enjoyment*. The number of trials (or error rate) and the task completion time (elapsed times) are used to measure the performance of the two interfaces.

We evaluated the performance and usability of the two interfaces for each query type (Sect. 4.2), using 10 test queries. The participants were instructed to formulate the given queries using both interfaces and fill out a questionnaire on the attitude criteria. During the experiments, we also measured the error rates and task completion times of the participants. The following hypotheses were determined before building the proposed hand interface.

H1 Spatial queries, where the user defines the object's position, size, and orientation, are easily formed using the mouse-based interface. We do not expect the hand-based interface to improve the usability and performance noticeably because the capability of a mouse is sufficient for these operations.

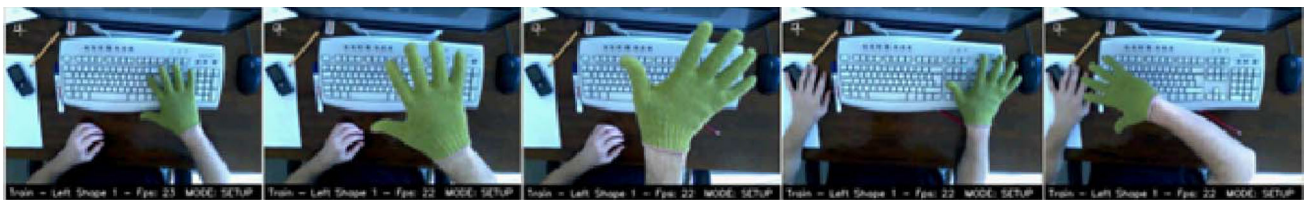

Fig. 9 Collecting training data for hand posture recognition: different sizes, orientations and positions of the same hand shape

Fig. 10 Some of the possible hand postures that can be used in a hand-based interface

H2 The main concern in a motion trajectory query is to draw a path. This can be easily handled by a mouse-based interface. We do not expect the hand-based interface to improve the usability and performance significantly over the mouse-based interface.

H3 Motion trajectory with temporal relation queries requires the correct timing of object motions relative to each other. As an example, consider a query to search for two objects colliding, or two cars moving from left to right, one after the other. Each object has its own path, and the paths should be drawn with the appropriate timing (temporal relation between motion trajectories). With a mouse-based interface, it is easy to draw the trajectory of one object, but hard to specify the timing for more trajectories. With a hand-based interface, it is easy to specify both the trajectory and timing of two objects using two hands. Therefore, we expect our hand-based interface to improve both the usability and the performance significantly.

H4 Camera motion queries define the parameters of a camera, such as its pan motion, zoom level, and rotation around the z-axis. Usually, when drawing a path for a pan motion, one must specify the zoom level of the camera during the pan motion as well. Since a mouse-based interface has a limited degree of freedom (x–y movement, right button click, etc.), it is not appropriate for specifying multiple camera properties at the same time. Unlike the mouse-based interface, a hand-based interface is better at providing many properties at once. For instance, the center position of a hand represents the position of a camera, while hand size represents its zoom level, and hand orientation represents its viewing angle. We expect that specifying camera motion queries using a hand-based interface is easier and more efficient.

6.1 Participants

A total of 12 volunteers (five female, seven male) participated in the study. Their average age was 36 years (*SD=6.2*), and their occupations were mathematician (1), physicist (2), system admin (1), secretary (1), computer scientist (2), student (3), and engineer (2). None of the participants reported previous experience with a hand-based interface or similar, and all of them were familiar with mouse interfaces.

6.2 Apparatus

The experimental setup consists of a desktop PC with a 1.7 GHz CPU of 2 GB RAM using Microsoft Windows XP, and a web camera with a resolution of 320 × 240 at a rate of 30 frames per second (fps). A tripod was used to locate the camera approximately one meter above the keyboard. Two uniformly colored cotton gloves (blue, green) of medium size were used for the hand-based interface. The system was implemented in C++ using the OpenCV library [15].

6.3 Procedure

Each participant was trained on the aim of the video retrieval system and how to formulate queries using the two interfaces. Then, they practiced several queries for each query type. This training took an average of 65 min (SD = 10) for each participant.

Each participant was instructed to formulate the 10 queries listed in Sect. 6.4. The query type and the order of interfaces (hand-based or mouse-based) were randomly selected. After performing the query on both interfaces, the participants filled out a questionnaire consisting of seven attitude criteria using a Likert scale from 1 (strongly disagree) to 5 (strongly agree). We also measured the elapsed time of the successful trials in seconds, and the number of trials to complete the query successfully. Since test queries contained several queries for some query types, we averaged their results (median for Likert scale, mean for ratio scale).

6.4 Test queries

Each user performed ten queries from the four query types using the mouse- and hand-based interfaces.

Type 1: Spatial queries

1. Two objects are side by side, with the left one smaller than the right one. They are facing each other.
2. Two objects are under one another, with the upper one facing right, and the lower one facing left.

Type 2: Motion trajectory queries

3. An object is moving from left to right on a linear path.
4. An object is moving from right to left on a sinusoidal path.

Type 3: Motion trajectory with temporal relation queries

5. Two objects are moving, with the left one moving from top to bottom, and the right one moving from bottom to top.
6. Two objects are colliding at the center of the scene. After collision, they move in opposite directions.

Type 4: Camera motion queries

7. The camera moves from left to right while zooming out.
8. The camera zooms in and then zooms out.
9. The camera pans right to left while moving left.

Fig. 11 Evaluation of spatial queries

10. The camera moves right while rotating from left to right and zooming in then zooming out.

6.5 Results and discussion

The experiment was conducted using a repeated measures within-subject design for two interfaces. For each interface, we collected seven attitude data and two objective performance data. Since attitude criteria are based on the Likert scale, we analyzed them using the binomial sign test, which is a nonparametric statistical test and used paired t test for the performance data, which are on a ratio scale. For all statistical tests, SPSS 17 was used, and the alpha level was set as .01 to test whether the differences were statistically significant.

6.5.1 Spatial queries (Type 1)

Our hypothesis (H1) states that there would be no difference between the two interfaces in terms of usability and performance. Figure 11 and the first rows of Tables 1 and 2 show the results of the attitude and performance criteria for spatial queries. Overall, our hypothesis is true since only natural and enjoyment criteria in the hand-based interface were rated higher than those in the mouse-based interface (see Fig. 11 and $p = .004$ (natural) and $p = .001$ (enjoyment) in the first row of Table 1). Table 2 indicates that the error rate and query completion time were not statistically different. This was an expected result due to the novelty effect of the hand-based interface. As a result, we can say that both interfaces are similar in terms of usability and performance.

Table 2 Results of paired-t test for error rate and elapsed time for the four query types

Type	Error rate		Elapsed time	
	$t(df = 11)$	Significance	$t(df = 11)$	Significance
1	−1.000	0.340	−2.690	0.020
2	0.560	0.580	0.290	0.770
3	−14.120	0.000*	−13.510	0.000*
4	0.430	0.670	−13.410	0.000*

* $p < 0.01$

6.5.2 Motion trajectory queries (Type 2)

Figure 12 shows the results of the motion trajectory query type, which are similar to the spatial query results in Fig. 11. Since the participants were familiar with mouse devices and the task is not difficult with a mouse, they did not find any noticeable differences between the two interfaces. These results confirmed that our hypothesis, H2, is true; however, participants indicated that the hand-based interface is more enjoyable to use than the mouse-based interface, $p = .004$, Table 1. This is also due to the novelty effect.

6.5.3 Motion trajectory queries with temporal relation (Type 3)

The results of the motion trajectory with temporal relation query type show significant differences between the two

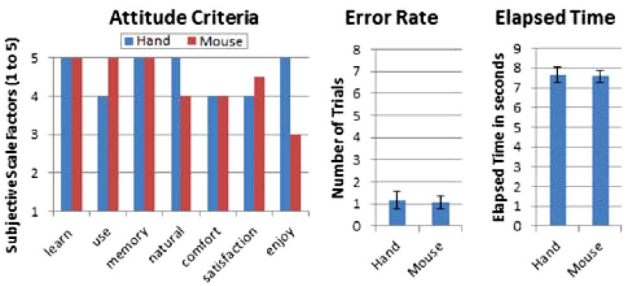

Fig. 12 Evaluation of motion trajectory queries

Table 1 Results of binomial sign test for attitude criteria for the four query types

Type	Significance (2-tailed)						
	Learn	Use	Memory	Natural	Comfort	Satis.	Enjoy.
1	0.625	0.016	1.000	0.004*	1.000	1.000	0.001*
2	0.500	1.000	0.680	0.040	0.070	0.125	0.004*
3	0.031	0.001*	0.031	0.001*	0.002*	0.001*	0.001*
4	0.039	0.001*	0.070	0.001*	0.109	0.008*	0.001*

* $p < 0.01$

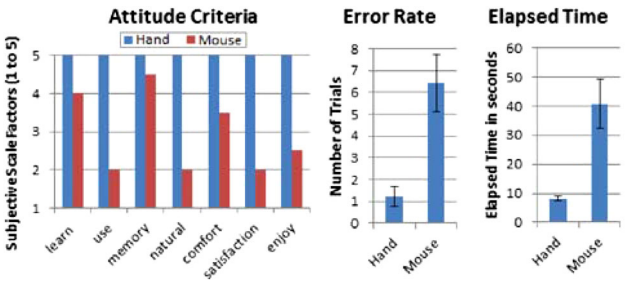

Fig. 13 Evaluation of motion trajectory queries with temporal relations

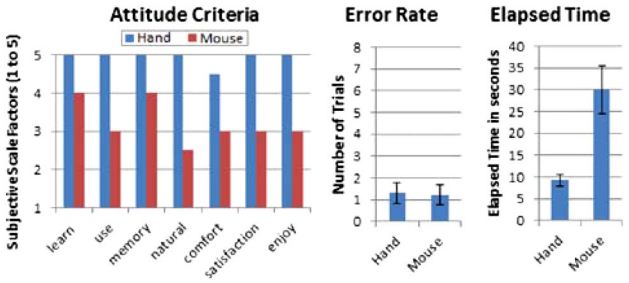

Fig. 14 Evaluation of camera motion queries

interfaces in terms of both attitude and performance evaluations. Specifying two paths while including temporal relations between two objects is very difficult in the mouse-based interface because controlling an object with a mouse and another with a keyboard requires good motor skills and practice. However, in a hand-based interface, the two objects in the scene are directly represented by the hands; it is easier to control one's hand movements when a device is not attached to them. Referencing Fig. 13 and Table 1, the hand-based interface is much easier to use ($p = .001$), more comfortable ($p = .002$), more satisfying ($p = .001$), more natural or familiar ($p = .001$), and more enjoyable ($p = .001$) than the mouse-based interface. These results indicate that participants found the hand-based interface as a more usable interface. The results of the objective performance criteria in Table 2 also show that the hand-based interface leads to lower error rate, and this difference is statistically significant: $t(11) = -14.12$, $p < .01$. The queries formed in the hand-based interface took significantly less time than those in the mouse-based interface: $t(11) = -13.51$, $p < .01$. Consequently, the hand-based interface outperforms the mouse-based interface in terms of both usability and performance criteria; these results support our hypothesis H3.

6.5.4 Camera motion queries (Type 4)

In the mouse-based interface, it is not possible to provide zoom levels and orientations while drawing the path of a camera. Therefore, the user must provide the camera properties in discrete time intervals. However, a hand as a device can represent many properties of a camera at once. For example, the center position of the hand shows the location of a camera, the distance of the hand from the camera indicates the zoom level, and the orientation of the hand represents the angle of camera around the z-axis. Therefore, while moving a hand, one can easily and continuously provide the zoom level and orientation of the camera. Using the hand directly eliminates the need for an external device and has the benefit of a higher degree of freedom. Since camera motion queries require the parameters of the camera continuously, the hand-based interface is more appropriate than the mouse-based interface. Figure 14 compares the two interfaces. The bar charts show that the hand-based interface is superior in both attitude and performance criteria. Participants indicated significant differences between the two interfaces in terms of use ($p = .001$), naturalness ($p = .001$), satisfaction ($p = .008$), and enjoyment ($p = .001$) in Table 1. Furthermore, Table 2 shows that participants completed camera motion queries using the hand-based interface faster than the mouse-based interface: $t(11) = -13.41$, $p < .01$. However, the error rate is not significantly different between the two interfaces because the participants made few mistakes and continuously setting the camera parameters took a long time. The results of the attitude and performance criteria indicate that the hand-based interface is more efficient and usable than the mouse-based interface; thus, our hypothesis, H4, is supported.

6.5.5 Performance of the visual processing module

Although our main focus was on evaluating the query interface rather than the low-level visual processing, we also evaluated the performance of the visual processing module to see whether it performs well enough so that the functionality of the query interface is not adversely affected due to problems in low-level processing. During each query performed by the participants, we collected the following data: (i) the number of frames in the query, (ii) the number of wrongly recognized hand postures, and (iii) the number of wrongly recognized hand-gestures. Then, the values are averaged over all queries. According to the results, one query takes about 8 s (242 frames at 30 fps) on the average, and only 2 % of the hand postures were recognized wrongly. The gesture recognition accuracy is 89 %; in other words, the system generates approximately one gesture command erroneously in ten queries. Using a low end PC with 1.7 GHz, we achieve about 20–22 frames per second. Our tests on a mid-range PC show that the system runs at 30 frames per second smoothly. These results show that under normal circumstances, the visual processing module can run in real time with high performance to support the query interface.

Our main goal is to determine whether a hand interface is better than the mouse–keyboard-based interface for video queries. We assessed mostly the subjective criteria; hence, the stability of the hand interface is crucial. A poor hand interface cannot compete with a stable mouse interface. Therefore, we gave higher priority to stability than flexibility (barehand). If we remove our restrictions (colored gloves for both hands), it leads to a more complex but fragile and less stable interface. The participant may have to repeat the same scenario many times due to the poor interface, and this adversely affects the subjective criteria. The orientation of thumb is a promising clue about the hand's type (left or right), but it still requires a good segmentation. Based on our tests, the color glove provides better results than barehand segmentation. Using only one colored glove instead of two relaxes the restriction, but it makes the case of overlapping hands harder. Wearing a long-sleeved shirt with the same color as the gloves fails our system, since hand regions are segmented based on color only. One possible solution would be to employ a wrist detection algorithm to segment and separate the hands from the arms; however, this would reduce the robustness and speed of the system.

7 Conclusion

We investigated whether a hand-based interface is appropriate for efficient query formulation on video retrieval systems. This was motivated by the observation that it is very hard to formulate certain queries with a mouse-based interface. Such queries usually require an interaction method/device with a higher degree of freedom; hence, the mouse is insufficient to formulate these queries efficiently. As a solution, we proposed a hand-gesture-based interface, by taking advantage of human hands as the main device for the interaction.

We conducted a user study to compare our hand interface with a mouse interface in terms of attitude and performance criteria on four types of video queries. The results showed that the participants were happy with the mouse interface for the formulation of spatial and motion trajectory queries and that a hand-based interface is more appropriate for motion trajectory with temporal relations and camera motion queries. For the latter queries, human hands are used as an interface device with higher degree of freedom than a mouse, which allowed the participants to prepare such queries efficiently.

Acknowledgments The authors would like to thank Rana Nelson for proofreading this manuscript.

References

1. Lew, M.S., Sebe, N., Djeraba, C., Jain, R.: Content-based multimedia information retrieval: state of the art and challenges. ACM Trans. Multimed. Comput. Commun. Appl. **2**(1), 1–19 (2006). doi:10.1145/1126004.1126005
2. Bastan, M., Cam, H., Gudukbay, U., Ulusoy, O.: BilVideo-7: an MPEG-7 compatible video indexing and retrieval system. IEEE MultiMed. **17**(3), 62–73 (2010)
3. Zhang, Z.: Microsoft kinect sensor and its effect. IEEE Multimed. **19**(2), 4–10 (2012)
4. Pavlovic, V.I., Sharma, R., Huang, T.S.: Visual interpretation of hand gestures for human–computer interaction: a review. IEEE Trans. Pattern Anal. Mach. Intell. **19**(7), 677–695 (1997)
5. Mitra, S., Acharya, T.: Gesture recognition: a survey. IEEE Trans. Syst. Man Cybern. Part C: Appl. Rev. **37**(3), 311–324 (2007)
6. Kolsch, M., Turk, M.: Fast 2D hand tracking with flocks of features and multi-cue integration. In: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, Workshop on Real-Time Vision for, Human–Computer Interaction, p. 158 (2004)
7. Segen, J., Kumar, S.: Shadow gestures: 3D hand pose estimation using a single camera. In: Proceedings of the IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, vol. 1, pp. 479–485 (1999)
8. Stauffer, C., Grimson, W.: Adaptive background mixture models for real-time tracking. In: Proceedings of the IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR), vol. 2, pp. 246–252 (1999)
9. Wang, R.Y., Popović, J.: Real-time hand-tracking with a color glove. In: ACM Transactions on Graphics (Proc. SIGGRAPH), vol. 28, no. 3, Article no. 63, 8 pp. (2009)
10. Dipietro, L., Sabatini, A.M., Dario, P.: A survey of glove-based systems and their applications. IEEE Trans. Syst. Man Cybern. Part C **38**(4), 461–482 (2008)
11. Numaguchi, N., Nakazawa, A., Shiratori, T., Hodgins, J.K.: A puppet interface for retrieval of motion capture data. In: Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Computer Animation. Eurographics Association, pp. 157–166 (2011)
12. Genç, S., Atalay, V.: Which shape representation is the best for real-time hand interface system? In: Proceedings of the International Symposium on Advances in Visual Computing: Part I, pp. 1–11. Springer, Berlin (2009)
13. Zhang, D., Lu, G.: Review of shape representation and description techniques. Pattern Recogn. **37**(1), 1–19 (2004)
14. Shackel, B.: Usability-context, framework, definition, design and evaluation. Interact. Comput. **21**(5–6), 339–346 (2009)
15. Bradski, G., Kaehler, A.: Learning OpenCV: Computer Vision with the OpenCV Library. O'Reilly Media Inc, Sebastopol (2008)