3D Interaction through a Real-time Gesture Search Engine

Shahrouz Yousefi and Haibo Li

KTH Royal Institute of Technology, SE-100 44, Stockholm, Sweden

Abstract. 3D gesture recognition and tracking are highly desired features of interaction design in future mobile and smart environments. Specifically, in virtual/augmented reality applications, intuitive interaction with the physical space seems unavoidable and 3D gestural interaction might be the most effective alternative for the current input facilities such as touchscreens. In this paper, we introduce a novel solution for real-time 3D gesture-based interaction by finding the best match from an extremely large gesture database. This database includes the images of various articulated hand gestures with the annotated 3D position/orientation parameters of the hand joints. Our unique matching algorithm is based on the hierarchical scoring of the low-level edge-orientation features between the query frames and database and retrieving the best match. Once the best match is found from the database in each moment, the pre-recorded 3D motion parameters can instantly be used for natural interaction. The proposed bare-hand interaction technology performs in real-time with high accuracy using an ordinary camera.

1 Introduction

Currently, people interact with the digital devices through the track pads and touchscreen displays. The latest technology offers single or multi-touch gestural interaction on 2D touchscreens. Although this technology has solved many limitations in human mobile device interaction, the recent trend reveals that people always prefer to have intuitive experiences with their digital devices. For instance, popularity of the Microsoft Kinect can demonstrate the idea that people enjoy experiences that give them the freedom to act like they would in the real world. However, when we discuss the next generation of digital devices such as AR glasses and smart watches we should also consider the next generation of interaction facilities. The important point is to select a suitable space and develop a technology for effective and intuitive interaction. An effective solution for natural interaction is to extend the interaction space from 2D surface to real 3D space [1,2]. For this reason, vision-based 3D gestural interaction might be hired to facilitate a wide range of applications where using physical hand gestures are unavoidable. Specifically, in future wearable devices such as Google Glass, 3D gestural interaction with augmented environments might be extremely useful. Therefore, developing an efficient and robust interaction technology seems to be a need for the near future. From technical perspective, due to the complexity,

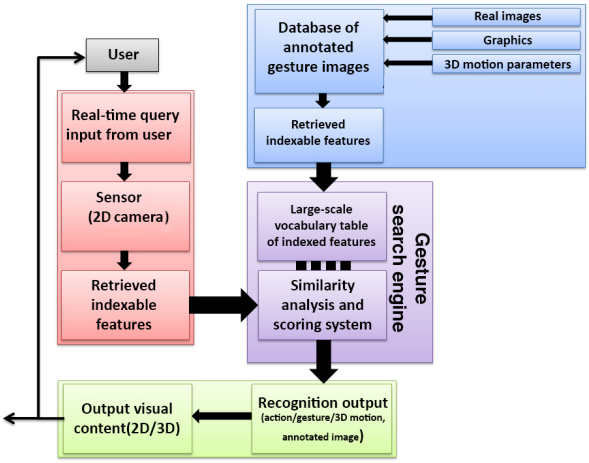

Fig. 1: System overview of the real-time gesture retrieval system. For each query image, the best corresponding match with the tagged motion information will be retrieved through the gesture search engine.

diversity and flexibility of the hand poses and movements, recognition, tracking and 3D motion analysis are challenging tasks to perform on hand gestures. In order to handle these difficulties, we decided to shift the complexity from classical pattern recognition problem to large-scale gesture retrieval system. Due to the possibility of forming a large-scale image database, the new problem is to find the best match for the query among the whole database. In fact, for a query image or video, representing a unique hand gesture with specific position and orientation of the joints, the challenging task is to retrieve the most similar image from the database that represent the same gesture with maximum similarity in position and orientation. Our matching method is based on the scoring of the database images with respect to the similarity of the low-level edge-orientation features to the query frame. By forming an advanced indexing system in an extremely large lookup table, the scoring system performs the search step and the best output result will be retrieved efficiently. Since in the offline step we annotate the database images with the corresponding global and local position/orientation of the joints, after the retrieval step, the motion parameters might be immediately used to facilitate the interaction between user and device in various applications (see Fig. 1).

2 Related Work

Designing a robust gesture detection system, using a single camera, independent of lighting conditions or camera quality is still a challenging issue in the field of computer vision. A common method for gesture detection is marker-based approach. Most of the augmented reality applications are based on marked gloves

for accurate and reliable fingertip tracking, [3,4]. However, in marker-based methods users have to wear special inconvenient markers. Moreover, some strategies rely on object segmentation by means of shape or temperature, [5,6,7]. Robust finger detection and tracking could be gained by using a simple threshold on the infrared images. Despite the robustness, thermal-based approaches require expensive infrared cameras which are not provided to most devices. Many gesture tracking systems are based on new depth sensors such as Kinect, but due to the size and power limitations they are only available for stationary systems [8].In addition, feature-based algorithms for gesture tracking have been employed in various applications, [7,9]. Model-based approaches are also being used in this area, [10,11].

Generally, all these techniques are computationally expensive, which is not suitable for our purposes. Another set of methods for hand tracking are based on color segmentation in appropriate color space, [5,12]. Color-based techniques are always sensitive to lighting conditions that degrades the quality of recognition and tracking. Other approaches such as template matching and contour-based methods often work for specific hand gestures, [13]. In new smartphones and tablets, accelerometer-based approaches recognize hand gesture motions by using the device's acceleration sensor, [14,15]. [16], use visual color markers for detecting the fingertips to facilitate the gesture-based interaction in augmented reality applications on mobile phones. [17,18], perform marker-less visual fingertip detection, based on the color analysis and computer vision techniques for manipulating the applications in human device interaction. [19], perform HMM to recognize different dynamic hand gesture motions. [20], use visual marker or shape recognition to augment and track the virtual objects and graphical models in augmented reality environments.

Unfortunately, most of the computer vision algorithms perform quite complex computations for detection and recognition of objects or patterns. For this reason we should find a totally innovative way to integrate the existing solutions with the minimum level of complexity and maximum efficiency. Another important point to mention is that the current technology is mostly limited to gesture detection and global motion tracking not real 3D motion analysis, while in many cases 3D parameters such as position and orientation of the hand joints might be used for manipulation in different applications. Therefore, besides the gesture recognition system we need to retrieve the 3D motion parameters of the hand joints (27 degrees of freedom for one hand). In our innovative solution we treat this issue as a large-scale retrieval problem. In fact, this is the main reason behind choosing very low-level features for efficient detection and tracking system. During the recent years, interesting works have been done on the large-scale image search topic. [21,22], perform the sketch-based image search based on the indexed oriented chamfer matching and bag-of-features descriptors, respectively. [23], introduces the matching based on distribution of oriented patches. The major problem with image search systems is that although you might receive interesting results in the first top matches but you also might find irrelevant results. Since our plan is to use the retrieval system for designing a real-time

interaction scenario, we expect to achieve around 100% correct detection and accurate 3D motion retrieval. In this work, we demonstrate that how our contribution leads to the effective and efficient 3D gesture recognition and tracking that can be applied to various applications.

3 System Description

As briefly explained before, our recognition and tracking system is based on the low-level edge-orientation features that can be achieved by hierarchical scoring of the similarity between the query and database images. Since hand gestures do not provide complex textured patterns, they are not suitable enough for detecting stable features such as SIFT or SURF. On the other hand, for robustness and efficiency of the detection and tracking, we cannot rely on color-based or shape-based approaches. These are the main reasons behind the selection of edge-based scoring system. As a result, the proposed method works independent of lighting conditions, variety of users, and different environments.

3.1 Pre-processing on the Database

Our database contains a large set of different hand gestures with all the potential variations in rotation, positioning, scaling, and deformations. Besides the matching between the query input and database, one important feature that we aim to achieve is to retrieve the 3D motion parameters from the query image. Since query inputs do not contain any pose information, the best solution is to associate the motion parameters of the query to the best retrieved match from the database. For this reason, we need to annotate the database images with their ground-truth motion parameters, P_{D_i}, and O_{D_i}. In the following we explain how the pre-processing on the database is performed.

Annotation of Global Position/Orientation to the Database: During the process of providing the database, one way to measure the corresponding motion parameters of the hand gesture is to attach the motion sensor to the user's hand and synchronize the image frames with the measured parameters. Another approach is to use computer vision techniques and estimate the parameters from the database itself. Since we could capture extremely clear hand gestures with a uniform background in the database, we could apply the second approach to estimate the global position of the gestures in each frame. As sample hand gestures are shown in Fig. 3, by using the common methods such as computing the area, bounding box, ellipse fitting, etc., we can estimate the position and scale of the user's gesture. On the other hand, to estimate the orientation of the user's gesture in x, y, and z directions, we apply *Active Motion Capture* technique [24,25]. In active motion capture, during the process of making database, we mount the vision sensor on the user's hand to accurately measure and report the motion parameters in each captured frame. The

vision sensor captures and tracks the stable SIFT features from the environment. Next, we find feature point correspondences by matching feature points between consecutive frames. Then the fundamental matrix for each image pair is computed using robust iterative RANSAC algorithm. Due to the fact that the matching part might be degraded by noise, the RANSAC algorithm is used to detect and remove the wrong matches(outliers) and improve the performance. Running RANSAC algorithm, the candidate fundamental matrix is computed based on the 8-point algorithm. The fundamental matrix F is the 3×3 matrix that satisfies the epipolar constraint:

$$x_i^{'T} F x_i = 0 \tag{1}$$

where x_i and $x_i^{'}$ are a set of image point correspondences. Each point correspondence provides one linear equation in the entries of F. Since F is defined up to a scale factor, it can be computed from 8 point correspondences. If the intrinsic parameters of the cameras are known, as they are in our case, the cameras are said to be calibrated. In this case a new matrix E can be introduced by equation:

$$E = K^{'T} F K \tag{2}$$

where the matrix E is called the essential matrix, $K^{'}$ and K are 3×3 upper triangular calibration matrices holding intrinsic parameters of the cameras for two views. Once the essential matrix is known, the relative translation and rotation matrices, t and R can be recovered. Let the singular value decomposition of the essential matrix be:

$$E \sim U diag(1, 1, 0) V^T \tag{3}$$

where U and V are chosen such that $\det(U) > 0$ and $\det(V) > 0$ ($\sim$ denotes equality up to scale). If we define the matrix D as:

$$D \equiv \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

Then $t \sim t_u \equiv \begin{bmatrix} u_{13} & u_{23} & u_{33} \end{bmatrix}^T$ and R is equal to $R_a \equiv UDV^T$ or $R_b \equiv UD^TV^T$. If we assume that the first camera matrix is $[I \mid 0]$ and $t \in [0, 1]$, there are then 4 possible configurations for second camera matrix: $P_1 \equiv [R_a \mid t_u]$, $P_2 \equiv [R_a \mid -t_u]$, $P_3 \equiv [R_b \mid t_u]$ and $P_4 \equiv [R_b \mid -t_u]$. One of these solutions corresponds to the right configuration. In order to determine the true solution, one point is reconstructed using one of four possible configurations. If the reconstructed point is in front of both cameras, the solution corresponds to the right configuration. Once the right configuration is obtained, the relative rotation between two consecutive frames are computed and can be tagged to the corresponding captured database image.

Annotation of Local Joint Motions to the Database: In order to annotate the local motion of the hand joints to the database we have used a semi-automatic

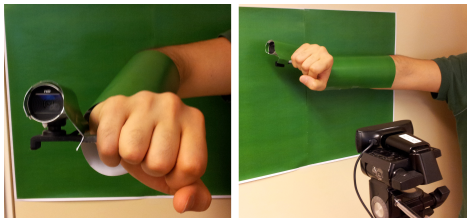

Fig. 2: Active motion capture setup for tagging the rotation parameters to the database images. The hand-mounted camera captures the global 3D rotation parameters and the static camera records the database frames. Both cameras are synchronized to automatically assign the real-time motion parameters to database frames.

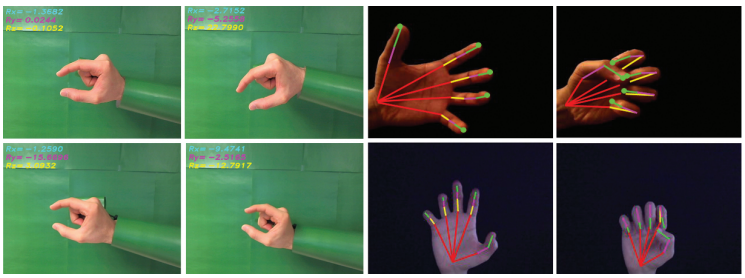

Fig. 3: Left: Real-time measurement of the global orientation of the hand gestures in the database images using Active Motion Capture system. R_x, R_y, R_z represent the rotation of the hand gesture around 3D axes in degrees. Right: semi-automatic annotation of the joint positions and skeletal model to the database images.

system. In this system we manually mark the fingertips and all the hand joints including the finger joints and wrist in each and every frame of the database. Afterwards, our system automatically stores the exact position of the marked points according to the image coordinates and generates the connection between the joints in form of a skeletal model. The joints information and hand model can be used after the retrieval step (see Fig. 3, right).

Defining and Filling the Edge-Orientation Table: Suppose that all the database images, D_{1-k}, are normalized and resized to mxn pixels and their corresponding edge images, ED_{1-k}, are computed by common edge detection methods such as Canny edge detection algorithm. Therefore, in each binary edge image, any single edge pixel can be represented by its row and column position. Moreover, it is possible to compute the orientation of the edge pixels, α_e, from the gradient of the image in x and y directions: $\alpha_e = atan(d_y/d_x)$. In order to simplify the problem, as it is demonstrated in Fig. 4 (top left), we divide the space to eight angular intervals, where the direction of each edge pixel belongs

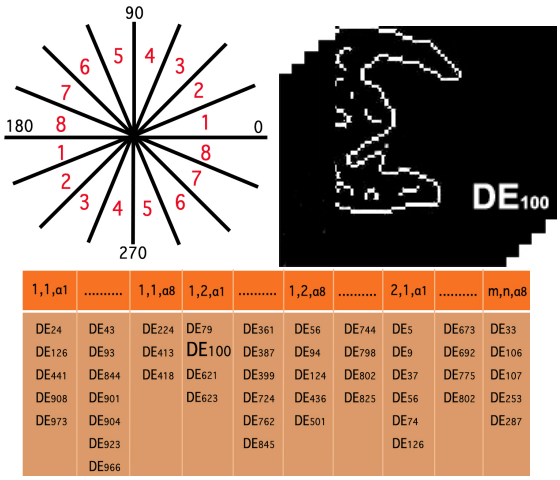

Fig. 4: Top-left: Associated angle intervals for edge pixels; Top-right: sample database edge image. The corresponding positions-orientation block to each single edge pixel will be marked with the index of the database image in the edge-orientation table.

to the one of these intervals. As a result, each single edge pixel will be represented by its position and angle: (x_e, y_e, α_e). In order to make a global structure for edge-orientation features we need to form a large table to represent all the possible cases that each edge-orientation pixel might happen. If we consider the whole edge database with respect to the position and orientation of the edges, (x_e, y_e, α_e), a large vector with size mxnxn_α, can define all the possibilities, where m and n are number of rows and columns in normalized database images and n_α is the number of angle intervals. For instance, for 320x240 images and 8 angle intervals we will have a vector with length, 614400. After we formed this structure, each (x_i, y_j, α_l) block should be filled with the indices of all database images that have edge at the same row, i, and column, j, with similar orientation interval, l. Fig. 4 shows how the edge-orientation table is filled with database images.

3.2 Query Processing and Matching:

The first step in the retrieval and matching process is edge detection. This process is the same as edge detection in the database processing but the result will be totally different, because for the query gesture we expect to have large number of edges from the background and other irrelevant objects. In the following we explain how the scoring system works.

Direct Scoring: Assume that each query edge image, QE_i, contains a set of edge points that can be represented by the row-column positions and specific

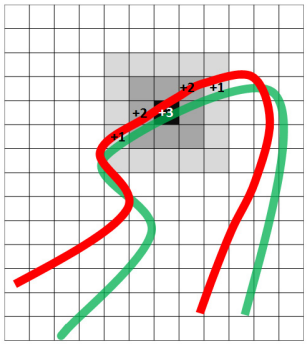

Fig. 5: The scoring process for a single edge pixel is depicted. Red and green patterns represent the database and query, respectively. Here, for the pixel marked with black, the associated scores for the red pattern with respect to the neighbor scoring are shown. The scores will be accumulated for the index of the corresponding database image. The same process will be done for all the edge pixels in the query pattern in comparison with all the database images.

directions. Basically, during the first step of scoring process, for all single query edge pixels, $QE_i \mid_{(x_u,y_v)}$, similarity function to the database images at that specific position is computed as:

$$Sim(QE_i, DE_j) \mid_{(x_u,y_v)} = \begin{cases} 1 \; if \; \{QE_i \mid_{(x_u,y_v)} \neq 0\} \\ \quad \wedge \{DE_j \mid_{(x_u,y_v)} \neq 0\} \\ \quad \wedge \{(\alpha_i \cong \alpha_j) \mid_{(x_u,y_v)}\} \\ \\ 0 \; otherwise \end{cases} \tag{5}$$

If this condition is satisfied for the edge pixel in the query image and the corresponding database images, the first level of scoring starts and all the database images that have an edge with similar direction at that specific coordinate receive +3 points in the scoring table. Similarly, for all the edge pixels in the query image the same process is performed and corresponding database images receive their +3 points. Here, we need to clarify an important issue that might be considered during the scoring system. The first step of scoring system satisfies our need where two edge patterns from the query and database images exactly cover each other, whereas in most real cases two similar patterns are extremely close to each other in position but there is not a large overlap between them (as demonstrated in Fig. 5). For these cases that regularly happen, we introduce the first and second level neighbor scoring. A very probable case is when two extremely similar patterns do not overlap but fall on the neighboring pixels of each other. In order to consider these cases, besides the first step scoring, for any single pixel we also check the first level 8 neighboring and second level 16 neighboring pixels in the database images. All the database images that have edge with similar direction in the first level and second level neighbors receive +2

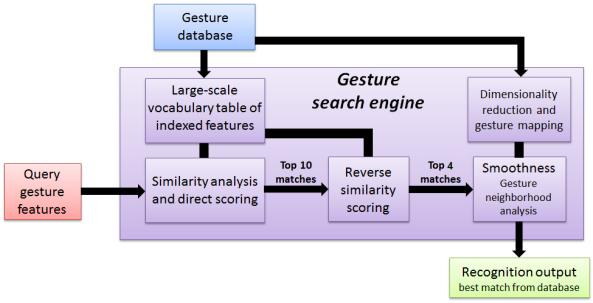

Fig. 6: Gesture search engine blocks in detail.

and +1 points respectively. In short, scoring system is performed for all the edge pixels in the query with respect to the similarity to the database images in three levels with different weights. Finally, the accumulated scores of each database image is calculated and normalized and the maximum scores are selected as first level top matches. The process of scoring for a single edge pixel is depicted in Fig. 5.

Reverse Scoring: In order to find the closest matches among the first level top matches, the reverse comparison system is required. Reverse scoring means that besides finding the similarity of the query gesture to the database images ($Sim(Q_i, D)$), the reverse similarity of the selected top database images to the query gesture should be computed. In fact, direct scoring system only retrieves the best matches based on the similarity of the query to them. This similarity might have happened due to the noisy parts of the query gestures. For instance, edge-orientation features of the background of the query image might be similar to a gesture database image. This similarity might cause the wrong detection. Therefore, similarity of the selected top database images to the query should be analyzed as well. Since database images are noise-free (plain background), similarity of the selected top matches to the query is a more accurate criterion. Combination of the direct and reverse similarity functions will result in a much higher accuracy in finding the closest match from the database. The final scoring function will be computed as: $S = [Sim(Q_i, D) \times Sim(D, Q_i)]^{0.5}$. The highest values of this function returns the best top matches from the database images for the given query gesture. In this work best top ten matches are selected in direct similarity. In reverse similarity analysis, best four database images of the previous step are selected. Afterwards, the smoothness process is performed to estimate the closest motion parameters for the query gesture image (see Fig. 6). Another additional step in a sequence of gestural interaction is the smoothness of the gesture search. Smoothness means that the retrieved best matches in a video sequence should represent a smooth motion. Basically, this process is performed in the following steps.

Weighting the Second Level Top Matches: In order to increase the accuracy of the 3D motion estimation, after the reverse scoring, we retrieve the tagged parameters from the four top matches and estimate the query motion parameters based on the weighted sum of them as follows:

$$P_Q = aP_{Dm_1} + bP_{Dm_2} + cP_{Dm_3} + dP_{Dm_4} \tag{6}$$

$$O_Q = aO_{Dm_1} + bO_{Dm_2} + cO_{Dm_3} + dO_{Dm_4} \tag{7}$$

Note that P and O represent the x-y-z tagged position and orientation, respectively. Q and Dm_i represent the query and $i-th$ best database match. Mostly, in the experiments, a, b, c, and d are set to 0.4, 0.3, 0.2, and 0.1. At this step the best motion parameters can be estimated for the first query in a video sequence.

Dimensionality Reduction for Motion Path Analysis: In order to perform a smooth retrieval, we analyze the database images in high dimensional space to detect the motion paths. Motion paths indicate that which gestures are closer to each other and fall in the same neighborhood in high dimension. The algorithm searches the motion paths to check which of these top matches is closer to the best found match for the previous frame. Therefore, if some of the selected top matches are not in the neighborhood area of the previous match, they should not affect the final selection and consequently the estimated 3D motion. For this reason, from the second query frame, the neighborhood analysis is performed and the irrelevant matches will be out from weighting the motion parameters.
For dimensionality reduction and gesture mapping different algorithms have been tested. The best achieved results that properly mapped the database images to visually distinguishable patterns are performed by Laplacian method. As demonstrated in Fig. 7, database images are automatically mapped to four branches. The direction of each branch shows the position of the hand gestures towards the four corners of the image frame. Clearly higher density of the points in the central part is due to the availability of the database images around the center area of the image frames. By using this pattern, from the second query matching, we can remove the noisy results. For instance, if one of the top four matches is out of the neighborhood of the previous match, it will be removed and weighing will be applied on the rest of the selected matches (see Fig. 7-left).
Another important point to mention is that if for any reason, the final top matches for the query frame are wrong (mainly due to the direct scoring), for the next frame the neighborhood analysis should not be considered. Otherwise the wrong detections significantly affect the estimated motion parameters. Therefore, if majority of the top four matches of the current frame are not from the neighborhood area of the previous match, they should be considered as a reference for estimating the 3D motion parameters and minority should be ignored from the computations (see Fig. 7-right).

Motion Averaging: Suppose that for the query images Q_{k-n}-Q_k $(k > n)$, best database matches are selected. In order to smooth the retrieved motion in

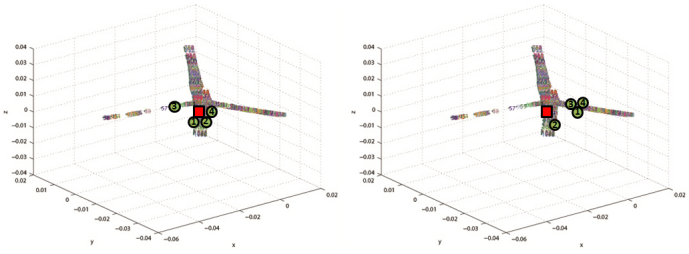

Fig. 7: Left: 3D motion estimation based on the top matches and neighborhood analysis. Red square indicates the best match from the previous frame. Numbered circles show the location of the top four matches for the current query frame. Based on the proposed algorithm, number 3 from the left plot and number 2 from the right plot should be ignored in the computations.

a sequence, the averaging method is considered. Thus, for the $k+1$th query image, position and orientation can be computed based on the estimated position/orientation of the n previous frames as follows:

$$P_{Q_{k+1}} = \frac{1}{n} \sum_{i=k-n+1}^{k} P_{Q_i} \tag{8}$$

$$O_{Q_{k+1}} = \frac{1}{n} \sum_{i=k-n+1}^{k} O_{Q_i} \tag{9}$$

Here, P_Q and O_Q represent the estimated position and orientation for the query images, respectively. Position/orientation include all 3D information (translation and rotation parameters with respect to x, y, and z axes). Therefore, motion parameters of each query image will be estimated by averaging the motion parameters of the certain number of previous image frames. According to the experiments, for $3 \leq n \leq 5$, averaging can be performed properly. For instance, if $n = 3$, motion averaging starts from the 4th query frame. 3D position and orientation of the 4th query frame will be estimated by the three previous frames and so on.

4 Experimental Results

The process of making database images and tagging the corresponding rotation parameters are implemented in C++. We synchronized two web-cams, one mounted on the user's hand to capture the hand motion and a static one to record the images for the database. Since the whole process is performed in real-time, the 3D hand motion parameters will be immediately tagged, as a separate text file, to each frame captured by the static camera. In order to provide extremely clear images for database, we covered user's arm and camera with similar paper

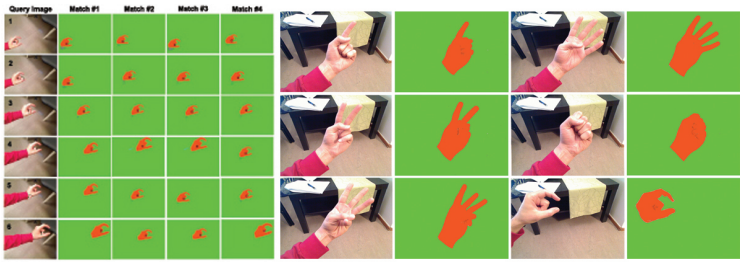

Fig. 8: Left: experimental results on a sample query video sequence of Grab gesture. The retrieved top four matches are shown on the samples. Right: different hand gestures and the corresponding best matches from the database.

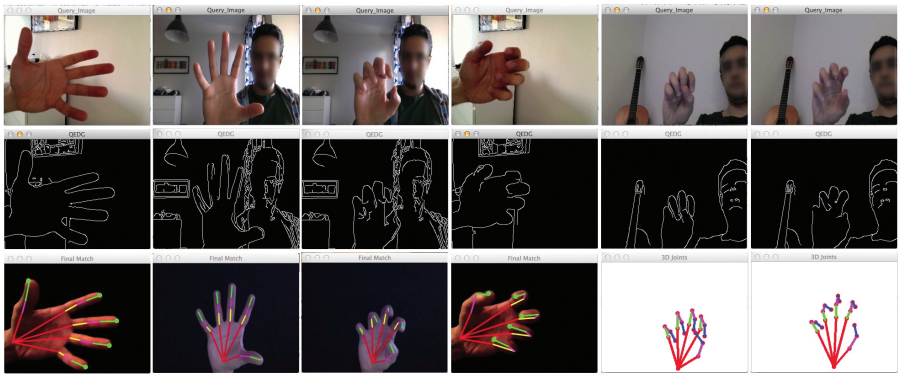

Fig. 9: First row: sample query frames from real-time video. Second row: detected edge from the query frames. Third row: corresponding best matches from the database with the annotated joint information.

to the background color. With some adjustments in the color intensity we could finally provide clear database images containing the user's gesture with a plain black or green background.
The matching experiments are conducted on different gesture databases. First, we provided the database with the specific hand gesture from a single user including all the variations in positioning, orientation and scaling (about 1500 images). During the second step, we extended the database to more than 3000 images of different dynamic hand gestures using one to five fingers and similarly including all the position/orientation variations. Finally, we added extra images to the database including the indoor and outdoor scenes, objects, etc. to test the robustness of the algorithm (totally more than 6000 images).

Our early experiments were conducted on a 2.93 GHz Core2Due PC. During the test step we used query gesture images from totally different environments with different backgrounds and lighting conditions. All the database images were organized in different scales for the experiments (320x240, 160x120, 80x60 and

few tests on 40x30). Obviously, the processing time is the feature that changes over the tests on different scales. We could reach reasonable processing time on the largest gesture database with size 320x240. The performance seems to satisfy the image-based retrieval at this level. Other important criteria to consider are the robustness in gesture recognition and accuracy in retrieving the 3D parameters. Our algorithm works with around 100% accuracy rate in recognition of the same gesture as the query even in the low-resolution case where we reduced the size of the database images to 80x60. We could achieve quite promising results in retrieving the 3D motion parameters up to the database images of size 160x120. In general the optimal point to achieve the best performance with respect to accuracy and efficiency is the test on gesture database with about 3000 entries with the image size of 320x240. We implemented the latest version of our system in Xcode environment on a Macbook Pro using the embedded camera. With this system we could easily achieve the real-time processing. The details about the performance of the system are depicted in Table. 1.
As discussed before, direct scoring, reverse scoring, weighting the top matches, and finally the motion averaging are the main four steps in estimation of the best motion information for the query image. During the direct scoring step top ten matches will be selected. Although many of these ten matches might be close enough to the query frame, but for accuracy reasons the best matches should represent the closest entries of the database to the query frame. Therefore, reverse scoring refines the top four from the previous step. Extending the reverse scoring to more entries can improve the final results but due to the efficiency reasons (reverse scoring substantially increases the processing time), this step is limited to ten top matches. Afterwards, we retrieve the annotated parameters from the first four top matches and estimate the query motion parameters based on the weighted sum of them. In cases that some entries are ignored due to the neighborhood analysis, weights will be allocated to the rest of the top matches. In general, reverse scoring and weighting system significantly improve

Table 1: Performance of the system with respect to the database size, image size, efficiency and accuracy.

Database size	Image size	Proc. time Sec.	Reco. rate	3D accuracy 1-5
6000($ges. + oth.$)	320x240	≈ 0.064	$\approx 100\%$	4
6000($ges. + oth.$)	160x120	≈ 0.049	$\approx 100\%$	3
6000($ges. + oth.$)	80x60	≈ 0.029	$\approx 100\%$	3
3000($ges.$)	320x240	≈ 0.048	$\approx 100\%$	4
3000($ges.$)	160x120	≈ 0.036	$\approx 100\%$	4
3000($ges.$)	80x60	≈ 0.027	$\approx 100\%$	3
1500($grabges.$)	320x240	≈ 0.033	$\approx 100\%$	4
1500($grabges.$)	160x120	≈ 0.025	$\approx 100\%$	4
1500($grabges.$)	80x60	≈ 0.016	$\approx 100\%$	3

the smoothness of the motion in a video sequence and remove the noisy results. In the final step, motion averaging is applied to enhance the fluctuations in the sequence of retrieved motion. Since the idea behind this work is to facilitate the future human device interaction in various applications, we should concentrate on effective hand gestures that might be useful in a wide range of applications. Based on the related works, the most effective hand gestures in 3D application scenarios are the family of Grab gesture [1,2] (including all dynamic deformations and variations) which is widely used in 3D manipulation, pick and place, and controlling in augmented/virtual reality environments. For this reason, these gestures are considered in most of our experiments while other hand gestures show the similar performance in the tests.

5 Conclusion and Future Work

In this work we proposed a novel solution for high degrees of freedom gesture recognition, tracking and 3D motion retrieval based on the gesture search engine. The proposed algorithm has successfully passed the inventive step and has been filed as a patent application (US Patent pending) in Jan. 2014. This method might be used in real-time gestural interaction with stationary or hand-held devices in a wide range of applications where gesture tracking and 3D manipulation are useful. Currently, we are implementing this technology on mobile devices to improve the quality of interaction in future applications. Here, an important point to mention is how to choose a reasonable size for the gesture database. Obviously, diversity and spatial resolution of the hand gestures are two main factors that directly affect the database size. In general, as discussed in [10], the hand motion has 27 degrees of freedom. Due to the correlation of the joint angles the dimension might significantly be reduced by applying dimensionality reduction techniques. In the current implementation, the vocabulary table can represent all possible indexable features that might occur (length of the search table is fixed). This indicates that the complexity of the processing does not depend on the size of the database and the current defined structures can handle substantially larger databases. Another important point is how to store the database. In fact, we do not need to store the database images. Instead, we only store the corresponding motion parameters and the search table. Size of the search table for database images of 320x240 is 614,400. According to our estimation each word in the search table will be marked with less than 100 entries of the database. Therefore, considering 2 bytes for storing each index of the database images in the search table we need around 100 MB of memory to store the whole search table. Obviously this amount of memory can be handled on any device. However, capturing, organizing and annotation of an extremely large database require substantial efforts which will be considered in the future work. Clearly, if we only target the gesture recognition and tracking, several thousand images are enough, but if we seek for high resolution 3D motion estimation we should increase the database size.

References

1. Yousefi, S.: 3D Photo Browsing for Future Mobile Devices (2012) Proceedings of the ACMMM12, October 29-November 2, Nara, Japan.
2. Yousefi, S.: Enabling Media Technologies for Mobile Photo Browsing (2012) Licentiate thesis, Department of Applied Physics and Electronics, Umea University, Umea, Sweden, ISBN:978-91-7459-426-3.
3. Dorfmueller-Ulhaas, K., Schmalstieg, D.: Finger Tracking for Interaction in Augmented Environments (2001) 2nd ACM/IEEE Int'l Symposium on Augmented Reality.
4. Maggioni, C.: A novel gestural input device for virtual reality (1993) In: Virtual Reality Annual International Symposium, IEEE, pp. 118 - 124.
5. Hardenberg, C.V., Berard, F.: Bare-hand human-computer interaction (2001) ACM International Conference Proceeding Series; Vol. 15 archive Proceedings of the 2001 workshop on Perceptive user interfaces, Orlando, Florida, Pages: 1 - 8.
6. Iwai, D., Sato, K.: Heat Sensation in Image Creation with Thermal Vision (2005) ACM SIGCHI Int'l Conf. on Advances in Computer Entertainment Technology.
7. Kolsch, M., Turk, M.: Fast 2D Hand Tracking with Flocks of Features and Multi-Cue Integration (2004) Proc. Computer Vision and Pattern Recognition Workshop.
8. Ren, Z., Meng, J., Yuan, J., Zhang, Z.: Robust hand gesture recognition with kinect sensor (2011) ACM Multimedia, pp.759-760.
9. Erol, A., Bebis, G., Nicolescu, M., Boyle, R., Twombly, X.: Vision-based hand pose estimation: A review (2007) Computer Vision and Image Understanding Volume 108, Pages 52-73.
10. Stenger, B., Thayananthan, A., Torr, P., Cipolla, R.: Model-based hand tracking using a hierarchical Bayesian filter (2006) IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(9):1372-84.
11. Yang, Sarkar, S.: Gesture Recognition using Hidden Markov Models from Fragmented Observations (2006) Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition.
12. Bencheikh, M., Bouzenada, M., Batouche, M.: A New Method of Finger Tracking Applied to the Magic Board (2004) Conf. on Industrial Technology.
13. Zhou, H., Ruan, Q.: Finger Countour Tracking Based on Model (2002) Conf. on Computers, Comunications, Control and Power Engineering, pp.503.
14. Arce, F., Valdez, J.: Accelerometer-Based Hand Gesture Recognition Using Artificial Neural Networks (2011) In Soft Computing for Intelligent Control and Mobile Robotics, Vol. 318, pp. 67-77.
15. Choi, J., Song, K., Lee, S.: Enabling a gesture-based numeric input on mobile phones (2011) IEEE International Conference on Consumer Electronics (ICCE), vol., no., pp.151-152.
16. Hrst, W., Wezel, C.: Gesture-based interaction via finger tracking for mobile augmented reality (2012) Multimedia Tools and Applications, pp. 1-26.
17. Baldauf, M., Zambanini, S., Fröhlich, P., Reichl, P.: Markerless visual fingertip detection for natural mobile device interaction (2011) Mobile HCI, p. 539-544.
18. Lee, D., Lee, S.: Vision-based finger action recognition by angle detection and contour analysis (2011) ETRI Journal, vol.33, no.3, pp.415-422.
19. Hannuksela, J., Barnard, M., Sangi, P., Heikkil, J.: Camera-Based Motion Recognition for Mobile Interaction (2011) ISRN Signal Processing.
20. Hagbi, N., Bergig, O., El-Sana, J., Billinghurst, M.: Shape Recognition and Pose Estimation for Mobile Augmented Reality (2009) 8th IEEE International Symposium on Mixed and Augmented Reality (ISMAR 2009), IEEE Computer Press.

21. Cao, Y., Wang, C., Zhang, L., Zhang, L.: Edgel index for large-scale sketch-based image search (2011) Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Colorado Springs, USA, isbn 978-1-4577-0394-2, p.761–768.
22. Eitz, M., Hildebrand, K., Boubekeur, T., Alexa, M.: Sketch-Based Image Retrieval: Benchmark and Bag-of-Features Descriptors (2011) IEEE Transactions on Visualization and Computer Graphics, vol.17, num.11,pp.1624 -1636.
23. Ikizler, N., Duygulu, P.: Human Action Recognition using Distribution of Oriented Rectangular Patches (2007) Workshop on Human Motion, pp.271–284.
24. Kondori, F.A., Liu, L.: 3D Active Human Motion Estimation for Biomedical Applications (2012) World congress on Medical Physics and Biomedical Engineering (WC2012), Beijing, China.
25. Kondori, F.A.: Human Motion Analysis for Creating Immersive Experience (2012) Licentiate thesis, Department of Applied Physics and Electronics, Umea University, Sweden, isbn: 9789174594164.