MASTER-PIECE: A Multimodal (Gesture+Speech) Interface for 3D Model Search and Retrieval Integrated in a Virtual Assembly Application

Konstantinos Moustakas[1], Dimitrios Tzovaras[1], Sebastien Carbini[2], Olivier Bernier[2], Jean Emmanuel Viallet[2], Stephan Raidt[3], Matei Mancas[4], Mariella Dimiccoli[5], Enver Yagci[6], Serdar Balci[6], Eloisa Ibanez Leon[7].

***Abstract*— The present report presents the framework and the results of Project 7: "A Multimodal (Gesture+Speech) Interface for 3D Model Search and Retrieval Integrated in a Virtual Assembly Application", which has been developed during the eNTERFACE-2005 summer workshop in the context of the SIMILAR NoE. The "MASTER-PIECE" (*M*ultimodal *A*ssembly with *SIMILAR *T*echnologies from *E*uropean *R*esearch utilizing a *P*ersonal *I*nterface in an *E*nhanced *C*ollaborative *E*nvironment) project aims at the generation of a multimodal interface using gesture and speech in order to manipulate a virtual assembly application. Besides assembling mechanical objects, the user is capable to perform 3D model content based search in a database of 3D objects using as query model a scene object. Finally, to deal with cases where no query model is available, a sketch based approach is proposed which results in the manual approximate generation of the query model. More specifically, the user can draw a specific number of primitive objects by moving his/her hands and then process-combine them so as to build more complex shapes, which are finally used as query models. Experimental results illustrate that the proposed scheme enhances significantly the realism of the interaction, while using the sketch-based approach the user can search for 3D objects in the database without the need of an initial query object, which is the case in most state of the art approaches.**

***Index Terms*— Multimodal interface, virtual assembly, 3D search, gesture, speech, sketch recognition.**

I. INTRODUCTION

During the latest years there has been an increasing interest in the Human-Computer Interaction society for multimodal interfaces. Since Sutherland's SketchPad in 1961 or Xerox' Alto in 1973, computer users have long been acquainted with more than the traditional keyboard to interact with a system. More recently with the desire of increased productivity, of seamless interaction and immersion, of e-inclusion of people with disabilities, as well as with the progress in fields such as multimedia/multimodal signal analysis and human-computer interaction, multimodal interaction has emerged as a very active field of research (e.g. [1] [2]).

Multimodal interfaces are those encompassing more than the traditional keyboard and mouse. Natural input modes are put to use (e.g. [3], [4]), such as voice, gestures and body movement, haptic interaction, facial expressions and more recently physiological signals. As described in [5] multimodal interfaces should follow several guiding principles: multiple modalities that operate in different spaces need to share a common interaction space and to be synchronized; multimodal interaction should be predictable and not unnecessarily complex, and should degrade gracefully for instance by providing for modality switching; finally multimodal interfaces should adapt to user's needs, abilities, environment.

A key aspect in multimodal interfaces is also the integration of information from several different modalities in order to extract high-level information non-verbally conveyed by users. Such high-level information can be related to expressive, emotional content the user wants to communicate. In this framework, gesture has a relevant role as a primary non-verbal conveyor of expressive, emotional information. Research on gesture analysis, processing, and synthesis has received a growing interest from the scientific community in recent years and demonstrated its paramount importance for human machine interaction (see for example the Gesture Workshop series of conferences started in 1996 and since then continuously growing in number and quality of contributions; a selection of revised papers from the last workshop can be found in [6]).

MASTER-PIECE integrates gesture and speech modalities into a designer and assembly application so as to increase the immersion of the user and to provide a physical interface and easier tools for design, than the mouse and the keyboard. Moreover, the user is capable of generating simple 3D objects and search for similar 3D content in a database, which is nowadays another very challenging research topic.

In particular, search and retrieval of 3D objects has application branches in numerous areas like recognition in computer vision and mechanical engineering, content-based search in e-commerce and edutainment applications etc. [7], [8], [9]. These application fields will expand in the near future, since

This report, as well as the source code for the software developed during the project, is available online from the eNTERFACE'05 web site: www.enterface.net.

1: Informatics and Telematics Institute, Centre for Research and Technology Hellas, Thessaloniki, Greece, e-mail: moustak@iti.gr, tzovaras@iti.gr
2: France Telecom R&D, Lannion, France, e-mail: sebastien.carbini@rd.francetelecom.com, olivier.bernier@francetelecom.com, jeanemmanuel.viallet@rd.francetelecom.com
3: INP-Grenoble, Grenoble, France, e-mail:Stephan.Raidt@icp.inpg.fr
4: Faculte Polytechnique de Mons, Mons, Belgium, e-mail: matei.mancas@tcts.fpms.ac.be
5: Universitat Polytechnica de Catalunya, Barcelona, Spain, e-mail: mariella@gps.tsc.upc.es
6: Bogazici University, Istanbul, Turkey, e-mail: enveryagci@yahoo.com, serdar.balci@boun.edu.tr
7: Technical University of Madrid, Madrid, Spain, e-mail: eloisa.i@caramail.com

the 3D model databases grow rapidly due to the improved scanning hardware and modeling software that have been recently developed.

The difficulties of expressing multimedia and especially three dimensional content via text-based descriptors reduces the performance of the text-based search engines to retrieve the desired multimedia content efficiently and effectively. To resolve this problem, 3D content-based search and retrieval (S&R) has drawn a lot of attention in the recent years.

However, the visualization and processing of 3D models are much more complicated than those of simple multimedia data [10], [11], [12]. The major difference lies in the fact that 3D models can have arbitrary topologies and cannot be easily parameterized using a standard template, which is the case for images. Moreover, there can be many different models of representing them, i.e. indexed facets, voxel models etc. Finally, processing 3D data is much more computationally intensive, than processing media of lower dimension, and often requires very large amounts of memory.

Many researchers worldwide are currently developing 3D model recognition schemes. A number of approaches exist in which 3D models are compared by means of measures of similarity of their 2D views [13], [14]. More direct 3D model search methods focus on registration, recognition, and pairwise matching of surface meshes [15], [16], [17]. Unfortunately, these methods usually require a computational costly search to find pairwise correspondences during matching.

Significant work has also been done in matching 3D models using geometric characteristics, where initial configurations are derived from conceptual knowledge about the setup of the acquisition of the 3D scene [18] or found automatically by extracting features such as curvature or edges [19]. When correspondences between the two objects to be matched are unknown, the registration problem, which in general is not well posed, may approximately be solved by the iterative closest point (ICP) algorithm [20]. In the absence of a priori knowledge or robust features, the ICP algorithm starts with one unique or, preferably, multiple different initial configurations [21]. In [22], a framework is presented for analyzing the subspace of the complete configuration space so as to force the ICP algorithm to converge to the global minimum. The method is evaluated experimentally for a number of real 3D objects.

A typical S&R system, like the aforementioned ones, evaluates the similarities between query and target objects according to low-level geometric features. However, the requirement of a query model to search by example often reduces the applicability of an S&R platform, since in many cases the user knows what kind of object he wants to retrieve but does not have a 3D model to use as query.

Imagine the following use case: The user of a virtual assembly application is trying to assemble an engine of its spare parts. He inserts some rigid parts into the virtual scene and places them in the correct position. At one point he needs to find a piston and assemble it to the engine. In this case, he has to manually search in the database to find the piston. It would be faster and much more easier if the user had the capability of sketching the outline of the piston using specific gestures combined with speech in order to perform the search. In the context of this project the integration of speech and gestures for the generation of the query model is addressed. Speech commands are used for performing specific actions, while gesture recognition is used to draw a sketch of the object and to manipulate the scene objects in the 3D space. The system is also capable of deforming objects and combine them so as to build more complex structures.

The rest of the document is organized as follows. Section II presents the general concepts of the developed application framework. In Section III the virtual assembly application and the 3D search engine are briefly described. Section IV describes the developed multimodal interface to the application and analyzes in detail all techniques used for the generation of the query model using sketches, while Section V presents the action verification module, which consists of a talking head. Finally, in Sections VI and VII two of the performed experiments are described and conclusions are drawn respectively.

II. Application framework

The developed application framework is a 3D assembly-designer interface. The user is capable to:

- Manipulate 3D objects in a 3D environment, including translation, rotation, scaling, etc.
- Assembly mechanical objects from their spare parts.
- Import 3D primitive objects using sketches.
- Manipulate and deform the primitive objects so as to generate more complex structures.

Unfortunately, an interface, like the predescribed one, is very difficult to use with standard keyboard-mouse input devices. The major problem stems from the fact that 3D actions can not be easily reproduced using 2D input devices.

The aim of presented framework is to add physical means of interaction between the user and the application and to overcome the need of the transition between the 2D input devices (mouse, etc.) and the 3D virtual environment. In particular, the developed multimodal interface consists of the modules:

- Speech recognition for specific commands.
- Gesture recognition to efficiently handle 3D objects using 3D motions of the hands.
- Recognition of sketches.
- Primitive models import and manipulation using gestures.
- Deformation of objects using gestures.

Finally, Figure 1 illustrates a block diagram of the developed multimodal interface

III. Basic modules

A. Virtual assembly application

The virtual assembly application is a graphical 3D interface for performing assembly of mechanical objects from their spare parts as illustrated in Figures 2a and 2b.

It has been initially developed to be used with haptic gloves and it allows the user to:

- Assembly a mechanical object from its spare parts.
- Grasp and manipulate objects using haptic gloves.

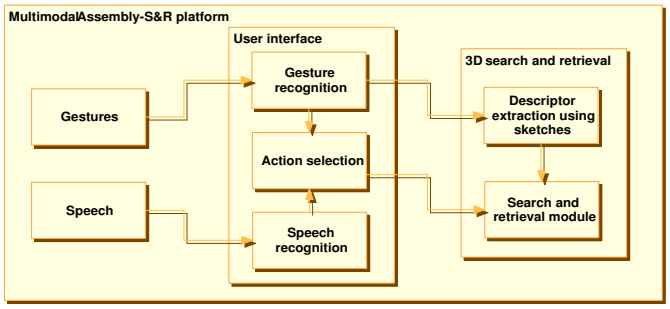
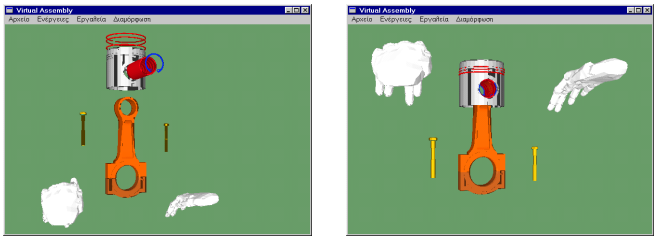


Fig. 1. Block diagram of the developed multimodal application



(a) (b)

Fig. 2. Virtual assembly application

- Author assembly scenarios using the authoring tool.

The user is also capable of assembling parts of an object and record the assembly process for post-processing. The assembly procedure can be done using one or two hands (i.e. one or two haptic VR gloves CyberTouch or CyberGrasp). A position tracker (MotionStar Wireless Tracker of Ascesion Technologies Inc. (2000)) with one or two position sensors installed is used to detect the position and orientation of the user hands in the space.

Another element of the application is the Virtual Reality (VR) agents module. VR agents are sophisticated software components with the capability to control a dynamic virtual environment and take actions based on a set of aims and rules. There are two kinds of agents implemented in the VR Assembly environment: a) the snap agent and b) the tool agents (a screwdriver and a wrench).

The snap agent is responsible to decide when two parts in the scene must connect. The aim of that agent is to place the components of the assembly in the correct position and to allow two or more components to construct a new larger component that can act like any other component. The rule that snap agent uses to connect two objects is a distance threshold and a "first contact rule". The "first contact rule" detects which sides of the objects collide first (using bounding boxes). If the colliding sides are valid, i.e. the objects are approaching from the sides that can snap, then the distance from the current position to the snapping position is calculated. When this distance is smaller than the distance threshold the objects snap to each other. The distance threshold used depends on the radius of the smaller bounding sphere of the objects.

The tools are components with the capability to control objects in the dynamic virtual environment. The tools aim to increase the immersion of the user in the VE. Unlike the snap agent, which is always active, the tools need to be activated by the user. Tools provide constraint to the object movement and allow the user do construction tasks. The virtual tools have the potential to increase user productivity by performing tasks on behalf of the user and increase the immersion of the user in the virtual environment. Furthermore use of virtual tools during assembly in the VE aids the user to detect possible construction difficulties related to the position and shape of the tools.

During the workshop the application has been extended to be used with a standard mouse-keyboard interface and also using the multimodal gesture-speech interface.

B. 3D content-based search engine

Using the 3D content-based search engine 3D objects can be retrieved from a database using another 3D object as query. Then the engine retrieves the most similar, in terms of a distance metric between their descriptor vectors, objects to the query model. The algorithm for extracting the geometrical descriptor of the object is briefly described in the sequel.

The extracted descriptors are rotation invariant. In particular, the object is initially normalized in terms of translation and scaling, i.e. it is translated to the center of the coordinate system, and is scaled uniformly so that the coordinates of all its vertices lie in the interval $[0, 1]$. Next, N_c concentric spheres are built centered at the origin of the coordinate system. Each sphere is built using tessellation of a normal icosahedron so that the vertices over its surface are uniformly distributed. In the experiments 20 concentric spheres of 16002 vertices are used. For each sphere the discrete 3D signal $F(r_s, \theta_i, \phi_i)$ is assumed, where i is the index of the sphere vertices. The values of function $F(r_s, \theta_i, \phi_i)$ are calculated using the Spherical Trace Transform (STT) [23].

The extraction of the final descriptor vectors, which is used for the matching algorithm, is achieved by applying the spherical functionals T, as described in [23], to the initial features $F(r_s, \theta_i, \phi_i)$ generated from the STT. The spherical functionals for each concentric sphere ρ are summarized below:

$$1.T_1(F) = \max\{F(r_s, \theta_i, \phi_i)\} \quad (1)$$

$$2.T_2(F) = \sum_{j=1}^{N_s} |F'(r_s, \theta_i, \phi_i)| \quad (2)$$

$$3.T_3(F) = \sum_{j=1}^{N_s} F(r_s, \theta_i, \phi_i) \quad (3)$$

$$4.T_4(F) = \max\{F(r_s, \theta_i, \phi_i)\} - \min\{F(r_s, \theta_i, \phi_i)\} \quad (4)$$

$$5.T_l(F) = A_l^2 = \sum_m \alpha_{lm} \quad (5)$$

where N_s is the total number of sampled points ($\eta_j, j = 1, \ldots, N_s$) at each concentric sphere, $l = 0, \ldots, L$ and $-l < m < l$. α_{lm} are the expansion coefficients of the Spherical Fourier Transform [24]:

$$\alpha_{lm} = \sum_{i=1}^{N_s} F(r_s, \theta_i, \phi_i) Y_{lm}(\eta_i) \frac{4\pi}{N_s} \quad (6)$$

where $Y_{lm}(\eta_i)$ corresponds to the spherical harmonic function, which is defined through:

$$Y_l^m(\theta,\phi) = k_{l,m} P_l^m(\cos\theta)\, e^{jm\phi} \quad (7)$$

where P_l^m is the associated Legendre polynomial of degree l and order m, $k_{l,m}$ a normalization constant and j the imaginary unit.

The quantities A_l^2 are invariant to any rotation of the 3D model. Choosing a sufficiently large number of L coefficients of the Spherical Fourier Transform, a total number of $L+4$ spherical functionals is used for each concentric sphere.

Finally, the descriptor vectors $\mathbf{D}(l)$ are created, where $l = 0,\ldots,(L+4)N_c$ is the total number of descriptors and N_c is the number of concentric spheres. In the experiments described in the sequel, $L = 26$ and $N_c = 20$ were chosen.

Figure 3 depicts the retrieved objects using as query the first model of each column.

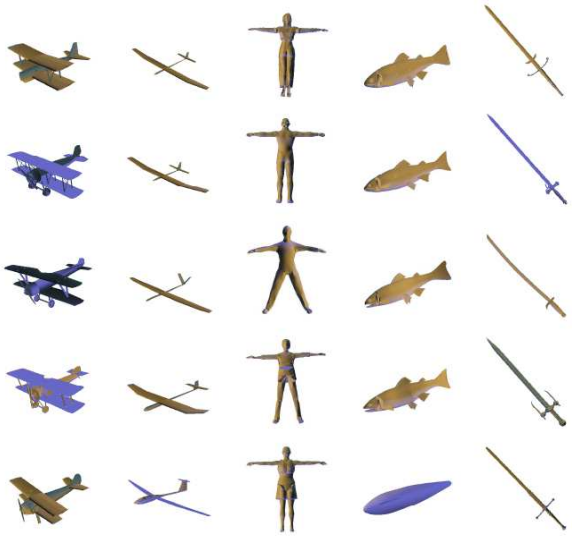


Fig. 3. 3D search results. The first row corresponds to the query object.

IV. Multimodal Interface to the Application

The developed multimodal interface to the application consists of two major parts. The interaction unit, which consists of the gesture-speech interface that enables the user to interact with the platform using only gesture and speech and the sketch-based query model generation system for the manual design of the model to search for. In the rest of this Section technical details about the algorithms used for the multimodal interface are presented.

A. Combined Gesture-Speech interface

1) Body parts detection and tracking: Most people instinctively use the eye-tip of the finger line to point at a target [25]. This convention is used in the present framework to estimate the pointing direction of the user [26], [25]. More precisely, the pointing direction is approximated by the head-hand axis.

Head and hands are detected and tracked [27] (Figure 4-left). Their positions in the 3D space are given by a stereo camera. The face of the user is automatically detected by a neural network more precisely described in [28]. The hands are detected as skin coloured moving zone in front of a vertical plane passing through the head, triggering pointing gesture recognition.

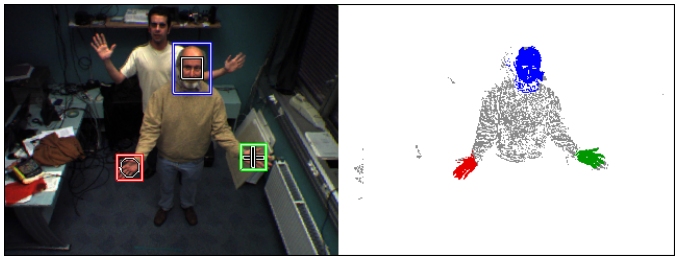


Fig. 4. Left: camera image (rectangle: head, circle: hand1, cross: hand2). Right: observations assigned to one of the four models depending on their probabilities (blue: head, red: hand1, green: hand2, grey: discard, white: pixels ignored in EM).

The first detected hand is tagged as "pointing hand" and the second detected hand is tagged as "control hand". The system can be used by right-handed persons as well as by left-handed persons (predominant hand is generally used to point) without differentiating explicitly right hand from left hand.

Once detected, the body parts (head and hands) are simultaneously tracked until tracking failure is automatically detected: then the detection for the lost part is re-triggered. The tracking process aims at explaining each new image by a statistical model with the EM algorithm [27]. The statistical model is composed of a colour histogram and a 3D spatial Gaussian function for each tracked body part (Figure 4).

2) Gesture interpretation: Gesture recognition is triggered by speech commands. The axis obtained from the first hand and the head 3D positions is used to compute pointed direction. The pointed direction is used when the user utters "selection" to select the pointed 3D object. Once selected, if the user utters "move", the object keeps moving to the pointed direction until the user utters "O.K.". When the word "rotate" is uttered, the current hand angles are taken as reference angles and current main axis of the object as reference axis. Then, until uttering "O.K.", the object rotates around its center of mass following the user's hands rotation according to the spherical coordinates (alpha and beta in Figure 5). When the user utters "scale", the current 3D distance between the hands is taken as reference value and current size of the object as reference size. Then, until uttering "O.K.", the object is continuously resized proportionately to the distance between hands (when distance between hands is two time higher than reference distance, the object is two time bigger than its reference size).

3) Speech recognition: To recognize speech commands, the speech signal is linearly sampled at 8 kHz in 16 bits. MFCC (*Mel Frequency Cepstrum Coefficients*) coefficients are computed, each 16 ms, on 32 ms signal frames. The recognition system uses the frame energy, 8 cepstral coefficients and an estimation of the first and second order derivatives of the speech signal. Thus, the observation vector has 27 dimensions.

The decoding system uses Hidden Markov Models. The

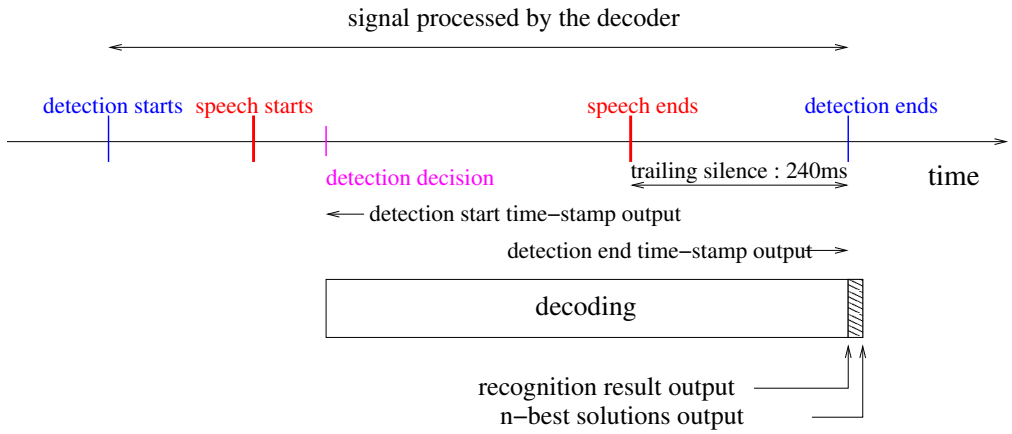


Fig. 6. Different times related to speech recognition. Speech recognition is only available after a delay following speech signal.

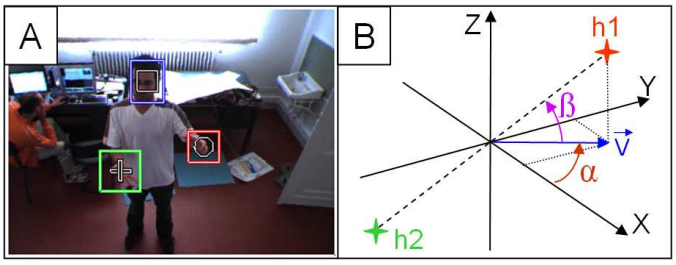


Fig. 5. (A): A user rotating a 3D object with his hands (blue: head, red: hand1, green: hand2). (B): the two hands h1 and h2 are defined by two angles α and β in the spherical coordinates.

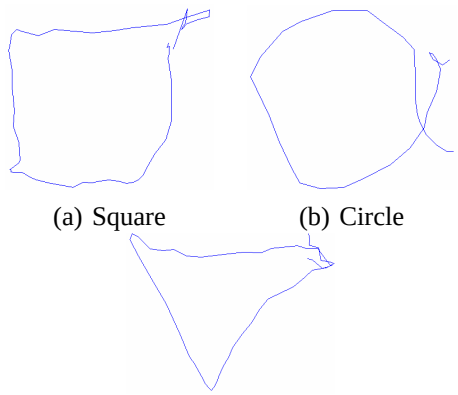
recognized sentences syntax is described in a grammar. The used vocabulary consists of 50 words. Dependent on the context, each word is obtained by phonetic units concatenation named allophones [29]. The system outputs the n-best results [30].

A noise/speech detector component filters the input signal to provide the decoder only with speech signal surrounded by silent frames. Beginning of detection is not causal. Detection component provides several frames which precede the speech detection decision. But as the decoder is faster than real time, it recovers from the non-causality of the detection component.

To detect end of speech, some consecutive silent frames must be observed. These frames are sent to the decoder. The best solution can be provided as soon as the last frame has been received. Computing the n-best solutions generate a negligible lag compared to the lag due to the silent frames. The number of frames to detect the end of speech is a parameter of the noise/speech component set to 15. The lag between the end of speech and the result of the recognition is thus 240 ms, since frames are grabbed each 16 ms. These times include the start and end silent frames which differ from the times of start and end of speech. These former can be computed from the noise-speech parameters. All the times constraints are summarized in Figure 6.

B. Sketch-based query interface

In this section the sketch recognition system will be described. Sketches are acquired via the gesture recognition module described in Section IV-A. The algorithms are designed only to recognize three basic shapes: square (or rectangle), circle and triangle. Figure 7 shows examples of traces obtained after acquisition.



(a) Square (b) Circle

(c) Triangle

Fig. 7. Examples of a square, a circle and a triangle after trace acquisition

It is obvious that these traces are far from perfection and often contain noisy points at the beginning and the end of the trace. Moreover, the point sets created by hand gesture recognition are generally distorted, while the sampling density is in general not constant.

The shape recognition systems can be grouped into two main categories. The first one could be described as feature based or rule based. Shapes are grouped using some human recognition based characteristic features as the number of corners, the number of parallel sides, etc. The second group consists of the model-based methods, which are based on fitting the different possible shapes on the trace to be recognized and to select the shape providing the highest correlation level with the trace to recognize.

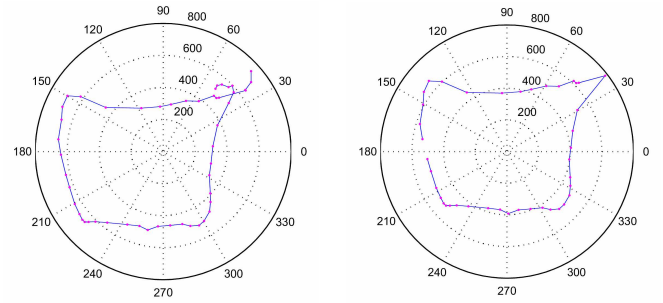
After describing the sketch denoising, the sketch recognition algorithms will be presented. Finally, possible extension of the sketch recognition system will be discussed.

1) Sketch denoising: As illustrated in Figure 7, some undesirable points occur in the acquired trajectory, especially at the beginning and the end of the trace. These points have to

be discarded to improve the results. Several approaches were designed and tested, both using directly the acquired points or transforming them into images. The result (discarding of noisy points) was evaluated in terms of correct recognition rate, robustness to noise, etc. and the main advantages and drawbacks are discussed at the end of this section. The sketch denoising methods can be divided into two more categories: those that only discard the meaningless points and those which also enhance and simplify the shape in order to make shape recognition more robust.

Method 1: Polar transform denoising

The first method discards some points considered as noise, which lie usually at the beginning and end of the drawing as can be seen in Figure 8. Points that follow in time according to their time stamp, but that do not correspond to a higher value of angle in the polar coordinate description, are deleted. This algorithm avoids also line crossings of the start and end points of the drawing.



(a) (b)

Fig. 8. Example of a square, before and after noisy points discarding using method 1

The main problems of this method are that we need to know which is the direction of the trace and which corner is the first one. So, extra algorithms are implemented to check that efficiently and aid the method.

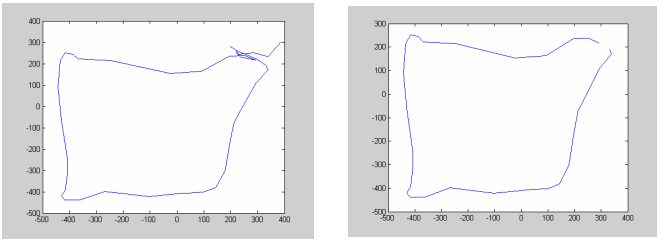
Method 2: Convex denoising

The second method searches for intersections in the tracked trajectory of the hand. The first and last points are investigated whether there is any cross section between any line segments. If the result is positive, the colliding edges are merged and the remaining parts of the trajectory (i.e. the line segments that do not belong to the closed contour) are discarded. Results are shown in Figure 9.

This method works well on all closed figures. If a figure is not closed as the example shown in Figure 10, it will fail.

Method 3: Statistical denoising

This method uses two statistical thresholds based on mean distance between two consecutive points. The first T_1 is used to remove small irregularities. In order to decide if N consecutive points constitute a small irregularity or not, a threshold is compared with the geodesic distance between the latest N+2 points. If the geodesic distance is bigger than a threshold value, which represents the weighted average distance between N points, we consider these points as noise and delete them.



(a) (b)

Fig. 9. Example of a square, before and after noisy points discarding using method 2

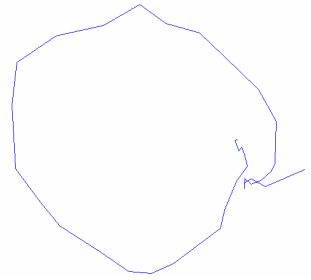
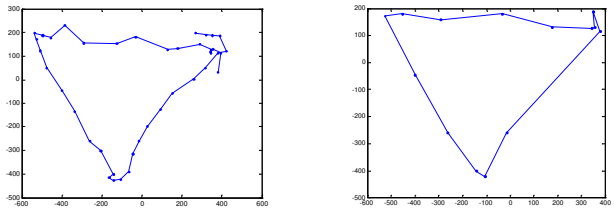


Fig. 10. Example of a circle, which is not closed

The second threshold T_2 is used to decide about the position where the trajectory should be closed so as to generate a closed contour. If there exists an intersection it is easy to decide about these position, as described in the previous methods. This method handles also cases of open trajectories. In particular, if the distance of the point considered, from at least one of the preceding points is smaller than T_2 these points are connected and the rest of the points outside the closed contour are discarded.



(a) (b)

Fig. 11. Example of a square, before and after noisy points discarding using method 3

In order to avoid connection of consecutive points and immediate failure of the algorithm, the preceding points of the considered one that are closer to it, in terms of geodesic distance, than T_1 are not considered for this test. Figure 11 illustrates results of this method.

Method 4: Image-based denoising

The fourth method converts the initial data set into an image by placing the linked trajectory points into an image raster. Then, image processing techniques are applied to process the traces. First of all a simple test is applied to test if the trace

is closed or not just by recursively evaluating the filled area of the trajectory: if the area is exactly of the same size as the image the trace is open, but if this area is smaller than the image, that means our trace was already closed. This method closes the trace if it was open before further processing.

In a second step the shape size is normalized and morphological opening is used in order to get rid of the meaningless points and to smooth the traces. As we previously normalized the shape size, it is possible to find a fixed kernel size for the morphological opening, which works better according to final visual check. Some results are shown in Figure 12, and this method worked well on all our experiments. Smoother shapes are obtained without the noisy points, which usually exist at the beginning and the end of the traces. This method works with both close or opened traces and it preserve the main features of the traces (corners, parallel sides, roundness). A possible drawback is the fact that filling big images could be a little long, but this problem could be solved by normalizing all traces at smaller sizes. Another drawback could be the fact that the point set is transformed into an image, so it would be better to use a recognition method also done in the image space. But it is still possible to convert the denoised image into a set of points in order to use a recognition method working directly with the set of points.

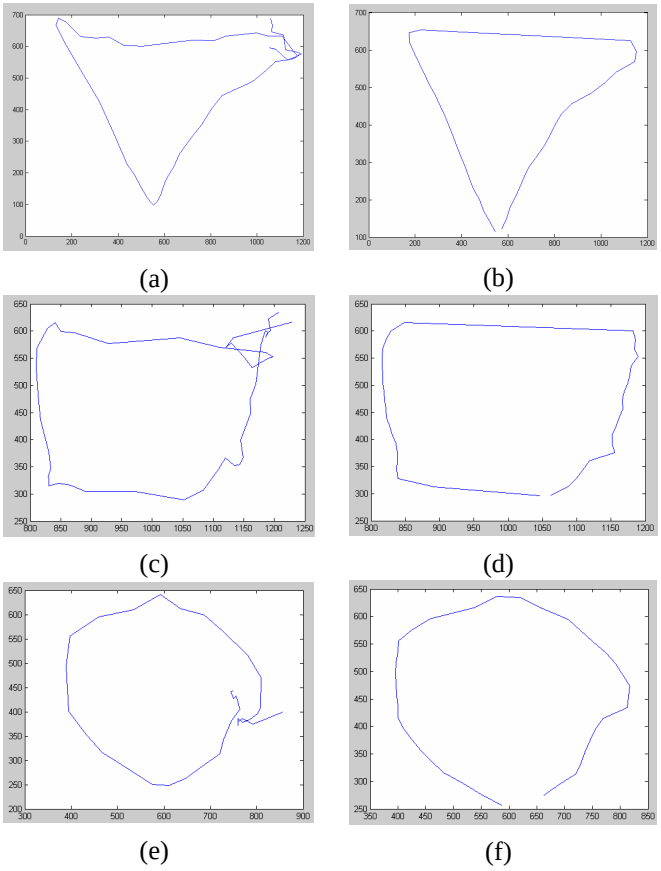


Fig. 12. Example of a triangle, square and circle, before (first column) and after (second column) noisy points discarding using method 4

2) Sketch recognition: The aim of the sketch recognition system is to recognize the shape of the trace and its main features (corners, center, radius, etc.) in order to draw then noiseless, squares, triangles and circles using the extracted features. The primitive-object vocabulary used demands the recognition of only 3 basic 2D shapes (triangle, circle or square). We tested several sketch recognition methods both in points set or image space.

Method 1: Polar transform recognition

In this method polar coordinates are used. In the first step of the recognition process the center of the shape is calculated as the mean of the X and Y coordinates. Measured points are usually not equally distributed so this center does not coincide with the real center of the shape but this problem is eliminated by using resampling, to get equally spaced sample points, and a denoising method as previously described.

At the beginning the distance between every point and the center is calculated and plotted into a "Distance-Angle" graph as illustrated in Figure 13. Theoretically, triangles should have three maxima/minima in this system, squares should have four (because of the three or four corners which are far away from the center) and circles may have many but very local maxima. For the triangle these peaks are equally spaced at 120^o and at 90^o in the case of the square. This interrelation is exploited for the recognition process. The detection of peaks and their angular distance gives sufficient information for the recognition comparing the calculated values to threshold values. The corresponding display of the example square in Figure 13a shows four distinct peaks. Ideally, these four peaks will be equally spaced and have the same amplitude, which is not the case here, due to noise, not exact sketching, etc.

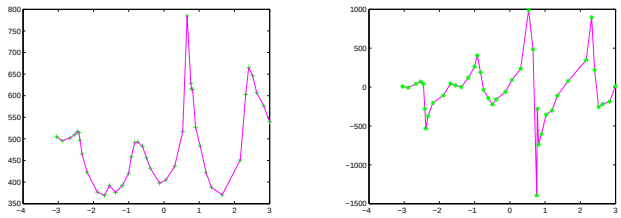


Fig. 13. a) Cartesian display of the polar coordinates (distance from center on the Y-axis, angle on the Y-axis), b)Result of differentiation of the distance rho

For making the system more robust it's been decided to differentiate the distance to the centre (rho) with respect to the angle (theta). This yields zero crossings where the peaks are with positive peak before and negative peaks after the zero crossing. This should allow easier localization of the peaks. The decision criteria of number of peaks and their distance remain. The circle should neither show distinct peaks nor equal distance between them, whereas their number may be very different. The result of differentiation is shown in Figure 13b.

Method 2: Vector-based recognition

The vector-based method takes advantage of the fact that the shapes to recognize have specific features that can be used for their recognition. I.e. lines consecutive points of a rectangle will have two main directions, of a triangle three, and for the circle many. The following assumption are considered in this algorithm:

1) Every rectangle consists of two sets of parallel lines.
2) Every triangle consists of three lines.
3) Circles cannot be separated into a specific finite number of lines
4) We have three different classes; triangle rectangle and circle and any given point set belongs to one of them.

This scheme is based on building a histogram of the direction of the line segments between two consecutive points, i.e. a histogram with the polar coordinate angles of each line segment. If the shape is a square then the histogram has two clearly defined maxima, corresponding to the directions of the two sets of parallel lines. If it is a triangle it should have three clearly defined maxima, while if it is a circle it has almost a uniform distribution. Using these features the task of classifying each point set into one of the three classes becomes trivial.

Method 3: Image-based recognition

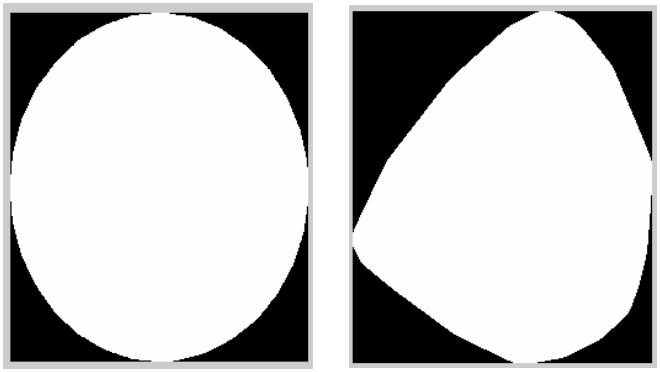
Another idea is to use the image space. After transposing the point set into an image, as previously described, the denoising method 4 is used to obtain proper filled shapes. Then, the bounding boxes are computed. Next, by comparing the area of the shapes to the bounding box area we can recognize them using the following procedure.

If the area is close to the bounding box the point set should represent a rectangle. If it is smaller it should correspond to a circle and finally if it is even smaller, it should be a triangle. An example is illustrated in Figure 14. At the left the shape bounding box and a perfect circle (an ellipse here) fitting into it is depicted. The right image illustrates the shape to classify. By comparing its area to the difference from the areas of the bounding box and of the ellipse it can be seen that the shape is more related to an ellipse than to a rectangle. For triangles the difference is more obvious since its area is much smaller than the bounding box and a simple threshold is enough to decided if the shape is a triangle or not. This method works very well once the thresholds are found and it ia also very fast.

In order to obtain a rotation invariant method, the rotation of the shape should be done until the bounding box area is minimum, but as only one parameter has to be optimized, the result is the global maximum and it is obtained fast without complex optimization methods.

Method 4: Recognition using Least-Squares optimization

The application developed for the 2D recognition of the noisy contours compares the least square errors for the geometrical shapes fitted to the data. The least square error for a geometrical shape with parameter vector $\mathbf{p}$, e.g. radius, side length, etc., is defined as the sum of minimum distances from the points in the curve to the geometrical shape, e.g.



(a) (b)

Fig. 14. Bounding box and fitting ellipse on the left, real shape on the right

$$F_p(x) = \frac{1}{2}\sum_{i=1}^{m}\left[f_i(x)\right]^2 \tag{8}$$

where $f_i(x)$ is the minimum distance of point i to the geometrical shape with parameter vector $\mathbf{p}$.

To find the optimum parameter vector $\mathbf{p}$ giving the minimum total error corresponding to geometrical shapes: circles, rectangles, triangles and ellipses; Levenberg-Marquardt nonlinear least squares minimization for unconstrained optimization is used [31], [32]. The geometrical shape giving the minimum sum of least square errors is identified as the best match for the contour drawn by the user. Vector $\mathbf{p}$ corresponding to the geometrical shape with minimum total square error is used to display the recognized shape to the user. The application for the 2D shape recognition consists of three major parts; namely, input of the contour information, functions calculating least square errors for various elementary geometrical shapes and the nonlinear least square algorithm.

The information for the contour drawn by the user is captured by the movements of the hands and is passed as a vector of points to the application. As the provided points are noisy and scattered, a filter is applied before the recognition algorithm is called. The filter resamples the points in the contour uniformly so that the distance between two successive points is kept constant. This prevents the erroneous effect of clustering of the points at the beginning and the end of the drawn contour. The filtered points are then passed to the shape recognition functions.

The second part of the application consists of functions calculating least square errors for elementary shapes such as circles, ellipses, triangles and rectangles. These functions mainly map a parameter vector $p \in R^m$ to an estimated measurement vector $\hat{x} = f(p), \hat{x} \in R^n$ where $n > m$. An initial parameter estimate p_0 is provided from the properties of the data, e.g. mean, max and min of the data points. The measurement vector is simply the distance of a point in the contour to the geometrical shape. The parameter and the measurement vectors are passed to the optimization function for finding the optimum parameters giving the minimum measurement error $\|e\|$ which s defined as:

$$\|e\|^2 = \|x - f(p)\|^2$$

where x is the measurement and $f(p)$ is its estimate.

As for the nonlinear least square method, Levenberg-Marquardt(LM) method is used for finding the parameters of the geometrical shapes. LM method is an iterative technique that finds a local minimum of a multivariate function that is expressed as the sum of squares of nonlinear functions. It can be thought of as a combination of steepest descent and the Gauss-Newton method. When the current solution is far from the correct one, the algorithm behaves like a steepest descent method: slow, but guaranteed to converge. When the current solution is close to the correct solution, it becomes a Gauss-Newton method. LM function calls the functions calculating the errors for principal shapes and returns the optimum parameters. The flow chart of the algorithm can be found in Figure 15.

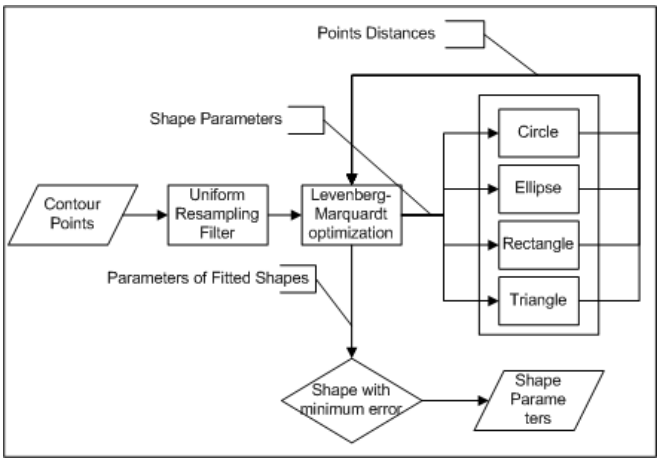


Fig. 15. Algorithm flow chart

The hand controlled mouse modality in this project allows the user to sketch contours in the Virtual Assembly Platform with the aim of creating 3D primitive geometrical shapes such as cylinders, square prisms, rectangular or triangular prisms from 2D sketches. The user draws a contour which is then recognized as a circle, a rectangle or a triangle. The recognized shape is used as a base for creating corresponding 3D shape such as circles for cylinders, rectangles for rectangular prisms etc.. The application for the extraction of 2D primitive geometrical shapes from the contours drawn with the hand controlled mouse modality has to be robust enough; because, the contours drawn by the user may be erroneous, incomplete and noisy; the points may be scattered, concentrated at the beginning or at the end of the contour or there may be unintentional lines drawn before the user stops sketching. Some contours drawn with the hand controlled modality are shown in Figure 16.

To evaluate the performance of the algorithm developed for 2D shape recognition, a database consisting of 100 contours is used where 98 shapes were detected correctly. The database is obtained by letting users to draw contours on the Virtual Assembly Platform using the hand controlled mouse modality.

All four sketch recognition approaches produced recognition rates close to 100%. For the final system method 4 was used due to its robustness, configurability and extensibility. It is not based on relative thresholds and not absolute ones which

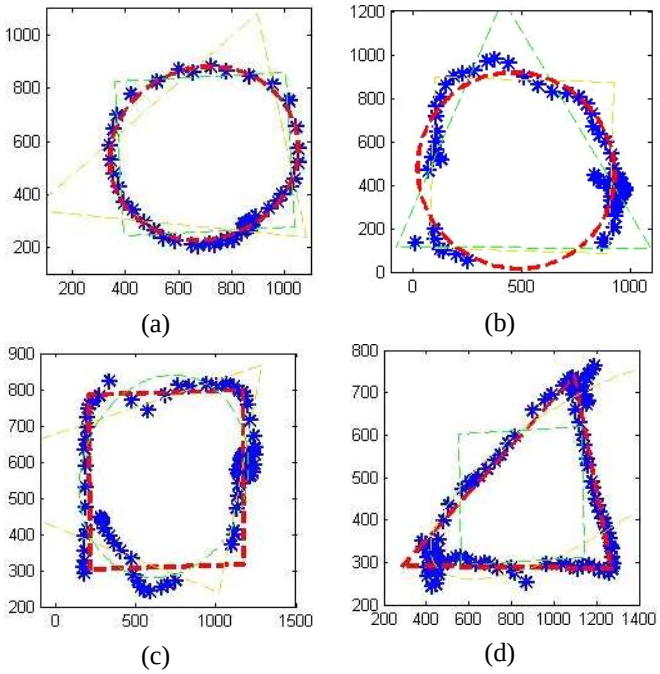


Fig. 16. a) A proper circular contour drawn by the user is shown in blue. The detected shape is drawn in red. b) An incomplete circular contour. c) A noisy rectangular shape d) A triangle with tails at the beginning and at the end

means that the results are more robust. Its only drawback for our case is the computation time. It was sufficient for our application but if real time sketch recognition is needed, the choice should be done on one of the methods 2 or 3 which are fast enough for real time. Moreover mix of different methods could be used.

3) Deforming the geometries: The range of objects, which can be built only by assembling primitives, is limited because irregular or non-symmetric shapes can not be drawn. Therefore the proposed 3D query model generation scheme using sketches integrates an object deformation procedure. It is implemented so as to interactively deform the initial 3D-primitive model generate a more complex one. The proposed geometric deformation technique, directly affects the triangulated model of a 3D-object.

Initially, a number of control points are defined on the surface of the object. In order to deform the mesh, the user has to translate its control points. This corresponds to setting a translation vector T_h for each control point. Translation T_h is propagated to the mesh and affects only the closest vertices to the control point. In particular it affects only the points $\mathbf{x}$ inside the geodesic window GW, which is defined as follows:

$$GW_u = \{x | \forall x \in V, g(u,x) < \varepsilon\} \tag{9}$$

where ϵ defines the window size, u is the control point and $g(u,x)$ the geodesic distance between x and u, i.e. the non-Euclidean distance on the surface.

All vertices lying inside the geodesic window are translated using the following equation:

$$\mathbf{T}(\mathbf{x}) = \mathbf{T}_h \cdot K(\mathbf{x}, \mathbf{u}) \tag{10}$$

where $K(\mathbf{x}, \mathbf{u})$ is a gaussian kernel used so as to assure smooth transition of the translation of the influenced vertices.

$$K(\mathbf{x}, \mathbf{u}) = e^{-\frac{g(\mathbf{x}, \mathbf{u})}{2\delta^2}} \quad (11)$$

where $\mathbf{x}$ is the position of the vertex and δ the standard deviation of the gaussian kernel. An important parameter which strongly affects the result of the deformation is the size ϵ of the geodesic window. Figures 17a and 17b illustrate the results of using two different values for ϵ. Notice that in the second case most of the points are deformed. In the context of the proposed framework the geodesic window is adjusted as the mean distance between two neighboring control points so as to make the user capable of deforming the whole surface of the object.

(a) $\epsilon = a$ (b) $\epsilon = 2a$

Fig. 17. Deformation using different values of ϵ

It should be also mentioned that in order to produce efficient deformations resampling is performed on the objects' surface so as to make it more regular and to assure smooth deformations.

4) Building the query-model: After the primitive objects are imported into the screen and processed accordingly (scaled, rotated, deformed) they are translated to the desired position so as to represent the targeting object. Next, they are grouped into a single object using speech commands. The resulting object is exported and its descriptor is extracted using the method described in Section III-B. Finally, the retrieved objects are imported into the scene, ordered in decreasing similarity, using speech commands.

V. Action verification module

To have feedback about the machine processing, a talking head was included. It provides audiovisual information about the recognized voice commands and the head and hand tracking.

The talking head used in this project was developed as a cloned 3D appearance with articulation gestures of a real human [33], [34]. The eye gaze and head orientation of the clone can be controlled independently to look at the user or where the user is looking on the screen. The virtual neck is also articulated and can accompany the eye-gaze movements. The audiovisual messages can either be recorded by the original human speaker, or synthesized from text input.

Most model-based and image-based systems describe the influences of movements like lip protrusion or jaw oscillation in limited regions of the face, whereas in reality articulatory movement produces deformations all over the face. For example the nose wings move during speech production and some

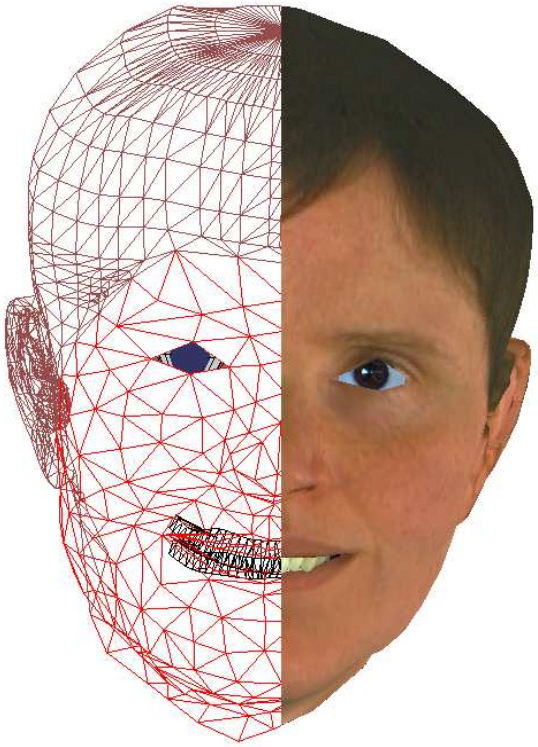


Fig. 18. The talking head

lingual and laryngeal movements have visible consequences on the cheeks and the throat. For the clone, it was an important issue to comprehend the influences by the subtleties of facial deformations induced by the underlying articulatory movements of visible speech. The facial animation is based on the recording of a French speaker, producing 40 prototypical configurations. His face was furnished with 166 colored beads. In a statistical analysis of the 3D points, six parameters could be identified that sufficiently model the facial movements of the lower part of the face.

For the multimodal interface the clone is used as a feedback of computer activity [35]. It aims to provide a more natural interface for the control of sketching and processing of shapes, that uses strategies humans use in face-to-face interaction. The aim of the clone as it is used in this application is to provide an appropriate feedback by the computer imitating the strategies a human might use. The clone greets the user when he or she is detected for the first time by the motion capture system. When the user is lost or localized again by the system, the clone makes corresponding utterances. During the sketching, when a hand is found and tracked, it follows the virtual pen drawing on the screen. Once the hand is lost or no hand has been localized it looks at the user's face. This provides information that a user can understand intuitively.

To confirm the recognition of voice commands, the clone announces the actions of the system that will follow. As some voice commands are used very often, such as for example "select", a random choice between several messages is provided, to avoid annoying repetitions of utterances by the

clone. In the current implementation, only French audiovisual speech synthesis for the clone was available.

VI. Application Demos

The proposed framework was tested in two scenarios, the assembly of a piston and the 3D content based search using as query a model generated using sketches and primitive objects.

A. Piston Assembly

In this scenario, the user had to assemble a piston using the developed gesture-speech interface. Specific speech commands described in Section IV-A.A were used to select an action and then the objects are manipulated using gestures. Figures 19a-d illustrate four consecutive steps of the procedure.

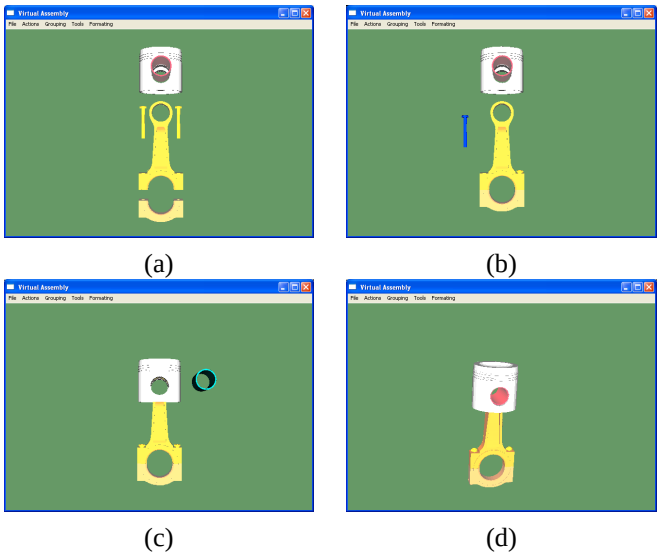


Fig. 19. Gesture-Speech based assembly of a piston

B. 3D content-based search using sketches

In this scenario the user has to draw primitive objects using sketches to assemble them into a more complex one and to search into the database for similar objects using the 3D content-based search engine (Figure 20).

Figure 21 illustrates four snapshots of the procedure of trying to design a car and to search for similar content. Notice that from the retrieved objects only the 8^{th} is not a car as illustrated in Figure 21d.

Figure 22 illustrates four snapshots of the procedure of trying to design a car and to search for similar content. Notice that all the retrieved objects are chairs 22d.

VII. Conclusions

In the present report we described the methods and results of Project 7 of the eNTERFACE-2005 summer workshop for multimodal interaction. The results of this work illustrate the efficiency of using multimodal interfaces to guide multimedia and VR applications. In particular, the gesture-speech controlled virtual assembly application has been observed to increase the immersion of the user to the application due to the physical interaction of the user with the system. Finally, the presented sketch-based query model generation for 3D search eliminates the demand of an existing query model in order to perform 3D search and provides the user with a natural means of generating in 3D the query object.

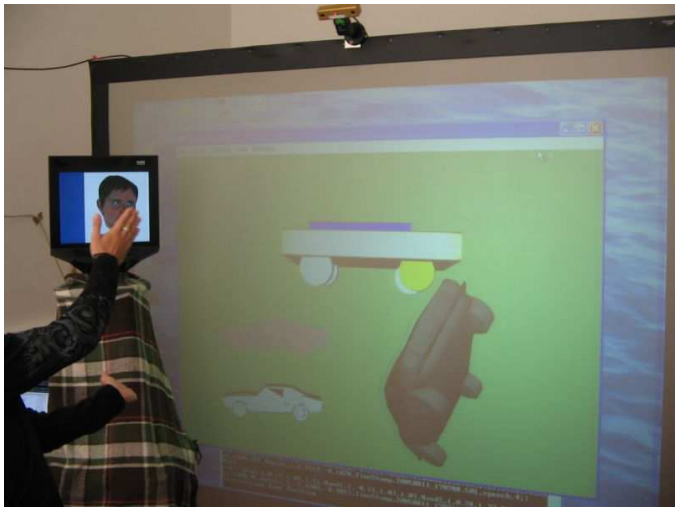


Fig. 20. Using the MASTER-PIECE platform - rotating a car

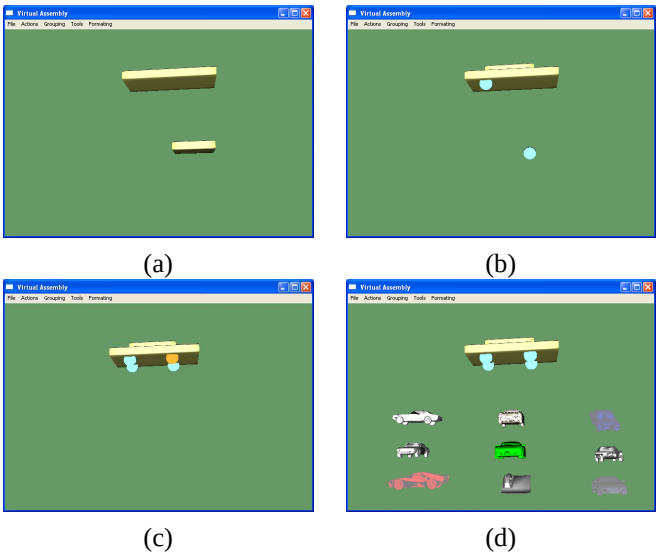


Fig. 21. Car sketching and recognition

VIII. Acknowledgements

This work was supported by the EU funded SIMILAR Network of Excellence.

References

[1] "W3c workshop on multimodal interaction," 19/20 July, 2004, Sophia Antipolis, France (http://www.w3.org/2004/02/mmi-workshop-cfp.html).

[2] "Special issue: Interacting with emerging technologies, j. strickon, guest ed., ieee computer graphics," Jan-Feb 2004.

[3] I. Marsic, A. Medl, and J. Flanagan, "Natural communication with information systems," in *Proc. of the IEEE*, 88, 8, Aug. 2000, pp. 1354–1366.

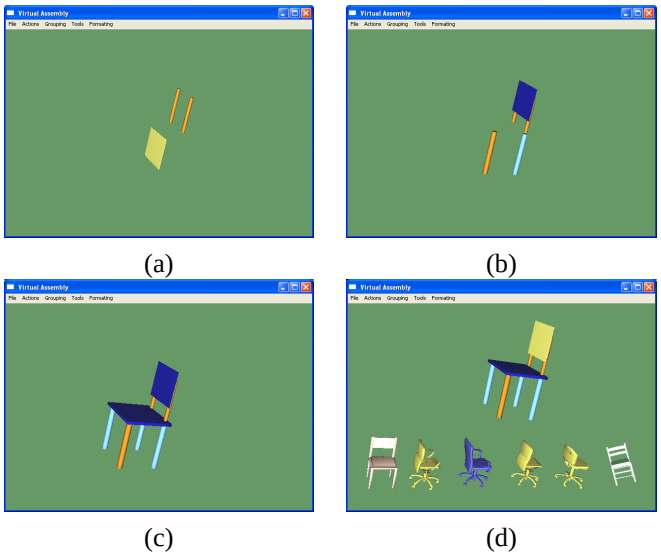


Fig. 22. Chair sketching and recognition

[4] J. Lumsden and S. A. Brewster, "A paradigm shift: Alternative interaction techniques for use with mobile & wearable devices," in *Proc. 13th Annual IBM Centers for Advanced Studies Conference CASCON'2003, Toronto, Canada*, 2003, pp. 97–110.

[5] T. V. Raman, "Multimodal interaction design principles for multimodal interaction," in *CHI 2003, Fort Lauderdale, USA*, 2003, pp. 5–10.

[6] A. Camurri and G. Volpe (Eds.), "Gesture-based communication in human-computer interaction," in *Lecture Notes in Artificial Intelligence, no. 2915, Springer Verlag, February*, 2004.

[7] S. Berchtold and H.P. Kriegel, "S3: Similarity search in cad database systems," in *Proc. of SIGMOD, J. Peckham, Ed. ACM*, 1997, pp. 564–567.

[8] H.P. Kriegel M. Ankerst, G. Kastenmuller and T. Seidl, "3d shape histograms for similarity search and classification in spatial databases," in *Proc. of the 6th Int. Symposium on Spatial Databases (SSD1999), Hong Kong, China*, 1999.

[9] U. Schurmans, A. Razdan, A. Simon, P. Mccartney, M. Marzke, D. V. Alfen, G. Jones, J. Rowe, G. Farin, D. Collins, M. Zhu, D. Liu, and M. Bae, "Advances in geometric modeling and feature extraction on pots, rocks and bones for representation and query via the internet," in *Computer Applications in Archaeology (CAA)*, 2001.

[10] E. Paquet and M. Rioux, "A content based search engine for vrml databases," in *Proc. of IEEE Conf. On Computer Vision and Pattern Recognition (CVPR1998)*, 1998.

[11] E. Paquet and M. Rioux, "Nefertiti: A tool for 3-d shape databases management," *SAE Transactions: Journal of Aerospace*, vol. 108, pp. 387–393, 2000.

[12] M. T. Suzuki, "A web-based retrieval system for 3d polygonal models," in *Joint 9th IFSA World Congress and 20th NAFIPS Int. Conf. (IFSA/NAFIPS2001), Vancouver*, 2001, pp. 2271–2276.

[13] J. Löffler, "Content-based retrieval of 3d models in distributed web databases by visual shape information," in *Proc. of Int. Conf. on Information Visualisation (IV2000)*, 2000.

[14] C.M. Cyr and B. Kimia, "3d object recognition using shape similiarity-based aspect graph," in *Proc. of Int. Conf. on Computer Vision (ICCV2001)*, 2001, pp. 254–261.

[15] D. Zhang and M. Hebert, "Harmonic maps and their applications in surface matching," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR99)*, 1999.

[16] G. Mori, S. Belongie, and J. Malik, "Shape contexts enable efficient retrieval of similar shapes," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR2001)*, 2001.

[17] A. E. Johnson and M. Hebert, "Using spin-images for efficient multiple model recognition in cluttered 3-d scenes," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 21, no. 5, pp. 433–449, 1999.

[18] G. Blais and M. Levine, "Registering multiview range data to create 3d computer objects," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 17, no. 8, pp. 820–824, 1995.

[19] C. S. Chua and R. Jarvis, "3d free-form surface registration and object recognition," in *Proc. of Int. Journal of Computer Vision, Kluwer Academic Publishers*.

[20] B. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 14, no. 2, pp. 239–256, 1992.

[21] H. Hugli, C. Schutz, and D. Semitekos, "Geometric matching for free-form 3d object recognition," in *ACCV, Singapore*, 1995, pp. 819–823.

[22] H. Hugli and C. Schutz, "Geometric matching of 3d objects: Assessing the range of successful configurations," in *Proc. of Int Conf. of Recent Advances in 3-D Digital Imaging and Modeling, Ottawa, Ontario, Canada*, May 1997.

[23] D.Tzovaras P.Daras, D.Zarpalas and M.G.Strintzis, "3d model search and retrieval based on the spherical trace transform," in *IEEE Intl Workshop on Multimedia Signal Processing, Sienna, Italy*, 2004.

[24] D.V. Vranic and D. Saupe, "Description of 3d-shape using a complex function on the sphere," in *Proceedings IEEE International Conference on Multimedia and Expo*, 2002, pp. 177–180.

[25] K. Nickel, E. Seemann, and R. Stiefelhagen, "3d-tracking of head and hands for pointing gesture recognition in a human-robot interaction scenario," in *IEEE International Conference on Automatic Face and Gesture Recognition*, Seoul, Korea, 2004, pp. 565–570.

[26] N. Jojic, B. Brumitt, B. Meyers, and S. Harris, "Detecting and estimating pointing gestures in dense disparity maps," in *IEEE International Conference on Face and Gesture recognition*, Grenoble, France, 2000, pp. 468–475.

[27] S. Carbini, J. E. Viallet, and O. Bernier, "Suivi statistique simultane des parties du corps pour des interactions bi-manuelles," in *ORASIS*, Fournol, France, 2005.

[28] R. Feraud, O. Bernier, J.E. Viallet, and M. Collobert, "A fast and accurate face detector based on neural networks," *Pattern Analysis and Machine Intelligence*, vol. 23, no. 1, pp. 42–53, January 2001.

[29] K. Bartkova and D. Jouvet, "Modelization of allophones in a speech recognition system," in *ICPhS (International Congress of Phonetic Science)*, Aix-en-Provence, France, 1991, pp. 474–477.

[30] R. Schwartz and Y.L. Chow, "The n-best algorithm: an efficient and exact procedure for finding the n most likely sentence hypothesis," in *ICASSP (International Conference on Acoustic Speech and Signal Processing)*, Albuquerque, USA, 1990, pp. 81–84.

[31] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly of Applied Mathematics*, vol. 2, no. 2, pp. 164–168, 1944.

[32] D.W. Marquardt, "An algorithm for the least-squares estimation of nonlinear parameters," *SIAM Journal of Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[33] F. Elisei G. Bailly, M. Berar and M. Odisio, "Audiovisual speech synthesis," *International Journal of Speech Technology*, , no. 6, pp. 331–346, 2003.

[34] G. Bailly L. Reveret and P. Badin, "Mother: a new generation of talking heads providing a flexible articulatory control for video-realistic speech animation," in *International Conference on Speech and Language Processing*, 2000, pp. 755–758.

[35] F. Elisei S. Raidt and G. Bailly, "Face-to-face interaction with a conversationnal agent: eye-gaze and deixis," in *International Conference on Autonomous Agents and Multiagent Systems, Utrecht, Netherlands*, 2005.

[36] PTGREY, "http://www.ptgrey.com," retrieved in 2005.

IX. Appendix: Software Notes

A. How to use: Assembly application

The virtual assembly application supports both mouse-keyboard and the multimodal gesture-speech interface.

For performing virtual assembly the user has to:

- Select one of the available assembly scenarios in "xml" format available in the "scenes" folder.
- From the "Action Menu" select "Start Assembly"
- Manipulate the objects, either using mouse-keyboard and the actions in the "Action Menu" or using gesture and speech as described in the following subsections
- When two objects that should be connected are close together they will snap.

For performing sketch-based 3D query model generation the user has to:

- Import the required primitive objects in the scene using either from the menu "File->Insert Primitive" or using sketches by "Actions->Start Sketching" or using gesture and speech as described in the following sections
- Manipulate the objects, either using mouse-keyboard and the actions in the "Action Menu" or using gesture and speech as described in the following subsections
- Group primitives into a single object: Use "Group->Select Group" to initiate grouping. Click on each sub-object and to "Group->Select Next" until all sub-objects are selected. Finally click on "Group->Finalize Selection". Alternatively, the corresponding gesture-speech commands can be used as described in the following subsections.
- Finally, use "File->Save Selected" to save the object and use "Actions->Search for Similar" to search for similar content.
- Only after the 3D descriptor extraction is terminated use "Actions->Retrieve Object" to retrieve the similar 3D objects from the database.

B. How to use: 3D search engine

The 3D search engine uses as input a 3D triangulated model in VRML format. The procedure for performing 3D search is demonstrated in the batch file preprocess.bat. In particular the user has to follow the next steps:

1) Generate the voxel model: Execute "$R3DST\_A$ name.WRL", where "name.WRL" is the name of the VRML file.
2) Generate initial descriptors: "$R3DST\_B$ name.vm K 16 3", where "name.vm" the name of the voxel model file, and "K", "16", "3" parameters of the algorithm.
3) Generate final descriptors: "$R3DST\_C$ name.Kraw_00 25", where "$name.Kraw\_00$" are the initial descriptors
4) Finally, file "$ST\_Descr.Kraw\_00.txt$" is generated, which contains the final descriptors.
5) Perform matching: "Retrieve.exe $ST\_Descr.Kraw\_00.txt$ DatabaseName", where DatabaseName is either ITI, PB or All corresponding in searching in the ITI the Princeton or both databases.
6) The retrieved object are listed in the resultsObject-Name.txt file where ObjectName is the name of the query model.

C. How to use: Gesture recognition

1) Presentation: The gesture program system needs a firewire stereo camera Bumblebee from Pointgrey [36]. It works under linux (Mandrake Linux 9.2 3.3.1-2mdk) and has been compiled with gcc version 3.3.1 on a Pentium IV 3Ghz computer.

The executable file <pointage> takes the images from camera, extract user body parts position (head and hands). Then, the direction pointed by the user and the angles between the hands are computed. These data can be sent through a pipe to another application or through the network using socket with the following format:

Head,Hb,Hx,Hy,Hz,Hand1,h1b,h1x,h1y,h1z,
Hand2,h2b,h2x,h2y,h2z,Pointer,px,py,
Angles,ab,a0,a1,TimeStamp,YYYYMMDD_HHMMSS.ms

With:

——- Head ———-

- Hb: head present boolean (true if the head is tracked).
- Hx,Hy,Hz: 3D position of the head in meters from the camera.

——- Hand1 ———

- h1b: first hand present boolean (true if the first hand is tracked).
- h1x,h1y,h1z: 3D position of the first hand in meters from the camera.

——- Hand2 ———

- h2b: second hand prsent boolean (true if the second hand is tracked).
- h2x,h2y,h2z: 3D position of the second hand in meters from the camera.

—— Pointer ——

- px,py: the location pointed by the user on the screen in pixels.

——- Angles ———

- ab: valid angles boolean (true if the two hands are tracked and if at least one of the hand is in front of user head)
- a0,a1: angles α and β of the hand.

——- TimeStamp ———

- YYYY: year MM:month DD:day HH: hour MM:minute SS:second ms: millisecond

For example when nobody is present the output of the system is:

Head,0,0.00,0.00,0.00,Hand1,0,0.00,0.00,0.00,
Hand2,0,0.00,0.00,0.00,Pointer,-1280,-1024,
Angles,0,0.0000,0.0000,TimeStamp,20050809_171000.106

2) Configuration and calibration: To configure the gesture application, the value of several parameters can be set in <rep_config/pointage.cfg>.

To configure the output socket or pipe:

- FLUX_SORTIE: Set to 1,2,3 to use a pipe to give data to another application on the same machine. Set -1 to use a socket with a TCP/IP connection to give data to a computer on the network.
- IPX_ADRESSE: four number for the ip adress of the remote computer when $FLUX\_SORTIE = -1$.

- PORT: port used when $FLUX\_SORTIE = -1$.

To calibrate the camera and the screen (figure 23), some parameters depending on their physical positions need to be set (all distances are in meters):

- DISTANCE_X_CAMERA_CENTREIMAGE: distance between the middle of the image and the camera on X axis.
- DISTANCE_Y_CAMERA_HAUTIMAGE: distance between the upper bound of the image and the camera on Y axis.
- DISTANCE_Z_CAMERA_IMAGE: distance between the screen and the camera on Z axis.
- ANGLE_CAMERA: angle between camera axis and Z axis.
- HAUTEUR_CAMERA: distance between the camera and the ground on Y axis.
- TAILLE_IMAGE_X, TAILLE_IMAGE_Y: horizontal and vertical size of the image.

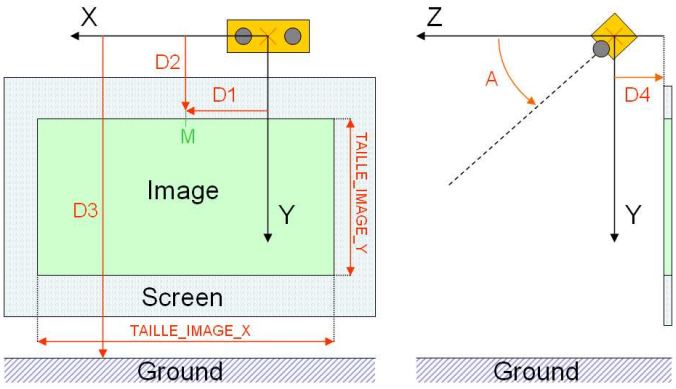


Fig. 23. Parameters for the camera and image: (D1) DISTANCE_X_CAMERA_CENTREIMAGE.
(D2) DISTANCE_Y_CAMERA_HAUTIMAGE.
(D3) HAUTEUR_CAMERA.
(D4) DISTANCE_Z_CAMERA_IMAGE.
(A) ANGLE_CAMERA.

Some parameters defines the limits of the interaction field (in meters):

- XE_MIN, XE_MAX: min and max X value for 3D coordinates.
- YE_MIN, YE_MAX: min and max Y value for 3D coordinates.
- ZE_MIN, ZE_MAX: min and max Z value for 3D coordinates.

The camera must be calibrated correctly.

- RESOLUTION_CAMERA_X, RESOLUTION_CAMERA_Y: work resolution of the camera (in pixels).
- B_EXPOSURE: exposure parameter for the camera.
- B_RED, B_BLUE: white balance setting (from 0 to 64).

The value of the parameters of the white balance depends on lighting condition. To find the good value for these two parameters (B_RED, B_BLUE) an automatic white balance procedure should be performed. A large white paper must be shown in front of the camera and $< W >$ key pressed on the keyboard while gesture application is running. The paper must represent the main part of the centre of the image. Once $< W >$ is pressed, the paper must be hold during about 10s. Then, in the console, the value for red and blue parameters are display in one of the last line (for example: $< --Valeursactuellesdelacamra : R : 13B : 47Exp : 400 >$ means that B_RED=13 and B_BLUE=47 are the correct value). The other parameters of the config file concern the detection and tracking of body parts and should not be changed.

D. How to use: speech recognition

The speech recognition use the executable files:
<../_paroleV4/iomshell>,
<../_paroleV4/bigfifo>
and <../_paroleV4/tst_mkvnb>
with the model:
<../modeles/echec_20050107_F15x27c_g08_R0p.gkz>
for the vocabulary definition. To test the speech recognition only, run the script $< reco >$. To change the number of n-best results given change the parameter $< -sol =>$. The first lines output when a word is recognize are:
$20050811_100711.858$ speech_start
$20050811_100712.083$ speech_end
!EXP: Src=si.inl;Type=f10.r10;File=5;

Then each n-best result is given on a separate line:
Sol=1 ; Dec="balou" ; Score=1155 ; Gscore=-60976 ;
NbFrames=38 ; Bnodes=81 ; Time=50 ;
Where <Sol=> is the rank of the n-best solution, <Dec=> contain the word recognized and <Score=> is the solution score.
The speech commands available are :

TABLE I
SPEECH COMMANDS

Nb	Speech Command	Action
0	no speech	no action
1	mougli	move
2	lâche	stop action
3	sélectionne	select object
4	balou	rotate object
5	chercanne	scale object
6	échec	search object
7	click	select group
8	prend la reine	select next object
9	tour prend	end select group
10	cavalier prend	retrieve an object
11	pose	save object
12	met le roi	delete object
13	met la reine	clone object
14	O.K. là	start sketching

E. How to use: fusion

The file <fusion> mix the speech recognition and the gesture recognition and send all to the 3D application via a socket connection. When gesture is mixed with speech, the parameter FLUX_SORTIE must be egal to 3 so that gesture application output results on a pipe to fusion application.
To change the IP adress of the computer where the 3D application is, change the value of ADRESSE_IP1 in the function void Acquerir().