The Hand As an Ultimate Tool

Ján Cíger, Department of Computer Graphics and Image Processing [*]
Jaroslav Plaček, International Laser Center, Bratislava [†]

Faculty of Mathematics and Physics, Comenius University, Bratislava

Abstract

This article describes a simple method, how to enhance human-to-computer communication using a tool, which is very natural to everybody – human hands. We are trying to create a system allowing the user to work intuitively and without unnecessary hassle. Such systems were traditionaly built with expensive hardware and special software. We are going the different way – commodity hardware and a bit of software "cheating". But nevertheless, the results are comparable.

Keywords: Virtual reality, motion capture, CCD cameras, image processing, gesture recognition

1 Introduction

The man used many tools in his every day's life during the ages. But only recently, the ultimate tool appeared – the computer. And right after pushing the computer out of the air-conditioned rooms onto the office desks for everybody's use as a personal computer, the need for an user friendly and intuitive method to use it, emerged. We can skip some decades from the prehistoric times to today and what we have there now – almost every personal computer is shipped with some kind of graphical user interface (GUI), all of them are based on a well known WIMP[1] model. History of this model was described extensively elsewhere and we will not repeat it.

But this model has one major design flaw – it is based on a desktop metaphor and uses a mouse as a way to manipulate objects laying on the desktop. This is all nice and cool, but how do most of the people work at their desks ? They use their hands to push papers, books and mugs of coffee forward and back and then write something with a pen, for example, and move the stuff on the desk again. Let's see how is this process done on the computer – we have to use the keyboard for writing. So far, so good – typewriters were in every day's use in many offices. But how to move the documents on the imaginary desk ? We have to use the mouse, pick something up and move it. And how to get to something laying under the pile of other stuff – we have to use the mouse again and clean everything out from the way, till we reach the needed object. On a regular desk, we can use both our hands to push the stuff away, to pick it up with one hand and get the needed paper from the bottom with the other hand, then put the pile back on the desk. This is not possible with a mouse, actually, using a mouse to do something like this on the computer screen is more like an attempt to drive a car with only an index finger of the left hand, than an intuitive and simple operation. And all these single and double clicks and all the "gymnastics" with the mouse is annoying a lot, especially for the computer newbies.

Our proposed solution is using a tool, which is very common and natural, instead of the mouse – the human hand. We are trying to create a system, which will recognize the postures and gestures of the user's hand and act upon them. This frees the way to enhancing the desktop metaphor to the almost "real" desktop. More elaborate comparison of human–to–computer interaction methods can be found also at [4].

2 Behind the scenes

The basic arrangement of the user's workplace can be as shown in the figure 1.

The system tracks the motion of the user's hands using the CCD camera. For simplicity and speed, we are not doing the complete motion-capture of the hand, but only the contours are tracked. The accuracy and completeness of tracking is only a minor issue for us, because this system is designed for interactive work and the user is always able to correct his actions when something is not tracked correctly. This limited amount of information is sufficient to allow the recognition of many simple gestural commands and movements of the hands. Such simplification enables the possibility to work without using any auxiliary markers or special equipment like data gloves.

This approach is not a new one – we are following the path shown in VIDEOPLACE [11], a revolutionary work combining art and completely new approaches to human–computer interaction.

Gesture recognition as way to control the computer is a rather popular approach, there are some related projects such as "Project BodyTalk" [1] and a military research presented at [6]. But both of them deal with recognition of only a small fixed set of gestures, associated to commands. Beside this, we are also trying to use hand movements and

[*] janoc@woc.sk
[†] jarop@ilc.sk
[1] WIMP - Window-Icon-Mouse-Pull Down

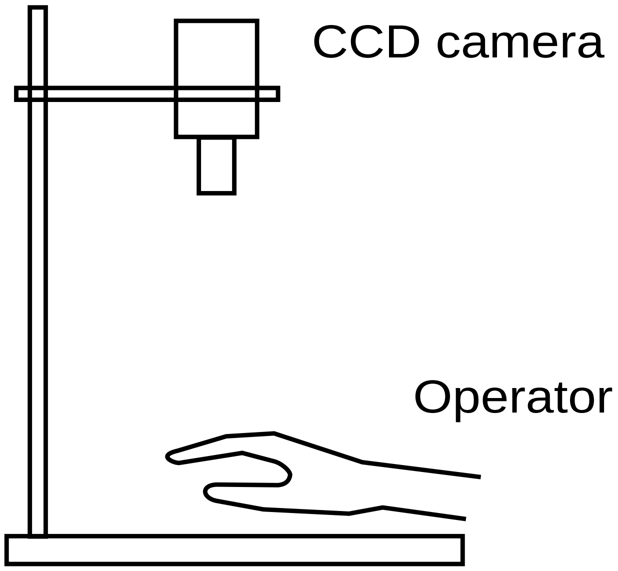

Figure 1: Basic arrangement

the exact positioning as a way how to be creative and more productive.

In the area of new interaction metaphors, there is an interesting project called "Studierstube/Virtual table" [2], but there are used only more "conventional" input devices like magnetic trackers.

The company NOVIA AG [7] offers the touchscreen in form of whiteboard which in cooperation with data projector simulates the mouse input. But this solution is also the aforementioned "one finger" approach.

The architecture of our system is showed in the figure 2. User's gestures are captured with the CCD camera. This data are used for direct feedback – the user sees the captured picture on his display. It has an advantage, that this feedback is extremely fast and natural for the user. This also enables the possibility to correct mistakes or results of bad tracking very quickly.

The next stage is classical image processing like filtering and thresholding. The final step in this stage is contour tracing and discovering of visible fingers and finger-tips by the one of the, below described, correlation or triangulation methods.

The preprocessed data are used for posture and gesture recognition. For the purpose of this project, posture is defined as a certain position of the hands and fingers in the scene. Gesture is defined as a temporal sequence of postures. Recognized postures and gestures are used for the further processing as described in the section 3. The obtained geometrical data are transformed using the calibration which takes into account the camera position and orientation and also objective distortions. The calibration has to be done during installation and whenever the camera position or orientation changes. It consists of capturing

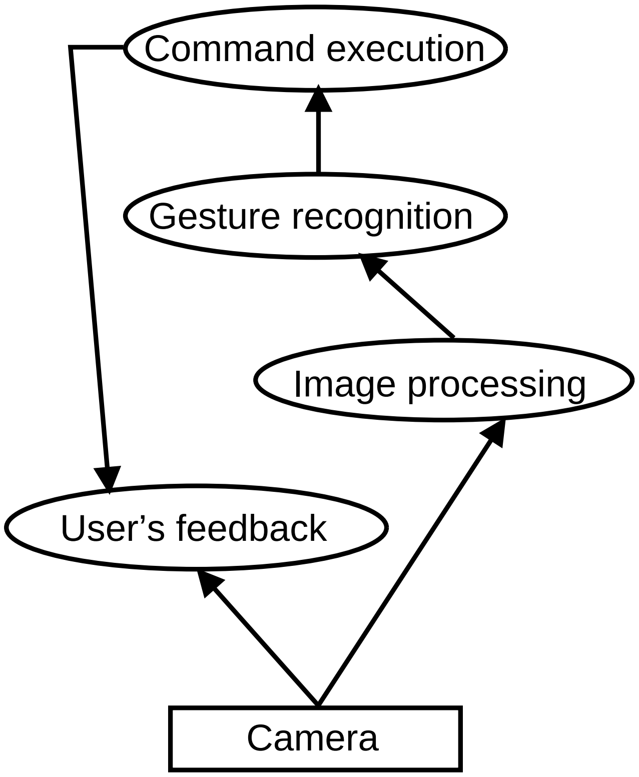

Figure 2: Architecture of the system

the reference grid and calculating the transformation for mapping the CCD camera image data to the application coordinate system.

3 Possible applications

Described system is in many cases suitable alternative to the standard input devices. This is the reason why we designed new applications on the basis of actually existing systems. When the system recognizes only one finger-tip it can be considered as the simulation of the touch-screen or the mouse. In case of recognizing the position of hands and fingers in 3D space you can attach the system output to the program which uses the data glove as the input. Here we introduce several possible applications ordered according to estimated complexity of implementation.

1. **Drawing tool for 2D.** Almost everyone, who works with computers, tried to sign himself using certain 2D drawing program. In most cases the result didn'l look good even if the author was a skilled veteran in the use of a computer mouse. Now lets think about drawing with finger in the sand. You can make really nice pictures even better than these made by pen, you can write your signature which will perfectly pass the authorization test and you will note, that such action can manage a child, four years old. The program with

the index finger acting as the brush and gestures replacing the menu shortcuts is not difficult to implement nevertheless it is a great improvement of standard painting applications interface (see Figure 3).

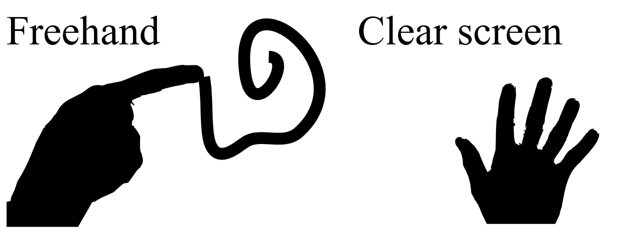

Figure 3: Concept of 2D drawing utility, open hand gesture clears the screen

2. **Desktop manager.** Imagine the desktop containing standard windows, menus, icons but no mouse cursor. How can you now manipulate with all the stuff there ? The function of cursor can be substituted by arbitrary of your ten fingers in dependency of actual gesture. The possibility of considering more than one cursor and detection of finger roots gives us the advantage over standard touch-screen. With index finger one can run programs just like with mouse but instead of double clicking you will tap on the icon, the gesture with all the ten fingers visible will clean the desktop and the files in the list can be selected just by the posture with stretched thumb and forefinger. Figure 4 illustrates the file selection and the resizing of the application window.

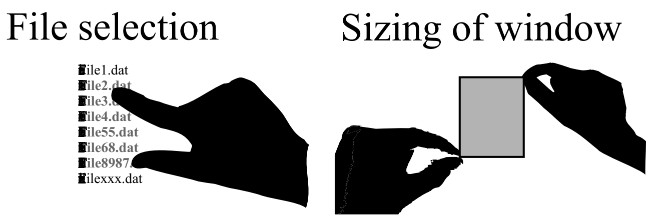

Figure 4: Control of the desktop manager

3. **Recognition of the finger alphabet.** The finger alphabet is a part of the sign language used for expression of the individual characters. It uses one hand postures which can be easily recognized using methods described later in the text. The simplicity of that alphabet consist of fact that it contains small number of characters defined by motion (American Sign Language Alphabet defines only characters I and Z by the gesture, all other are postures). In the system adaptation phase all the sample postures are assigned to alphabet letters. In the operating mode the system translates hand postures images to strings of characters. Figure 5 is an example of the postures representing letters B, Q, L, M in the German finger alphabet [5].

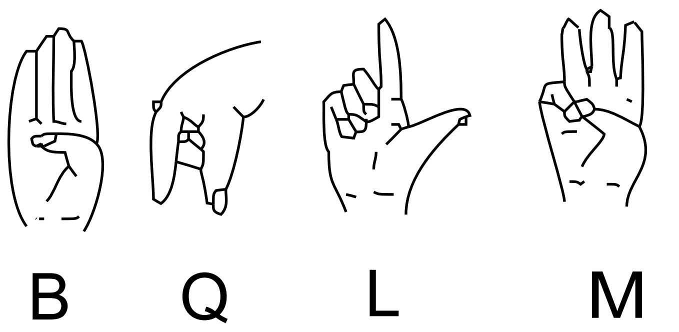

Figure 5: Examples of postures from finger aplhabet

4. **Modeling tool for 3D.** Programs for creation of 3D models and worlds usually use 2D input devices (mouse, tablet). Several applications were designed to use the virtual reality environment, but the standard VR hardware using data gloves and HMD's[2] interconnected by cables, causing the operator to feel like a dog on a leash. It certainly doesn't satisfy the requirements for a comfortable work. With use of the two CCD cameras the posture recognition system can provide the data which can replace the data glove output.

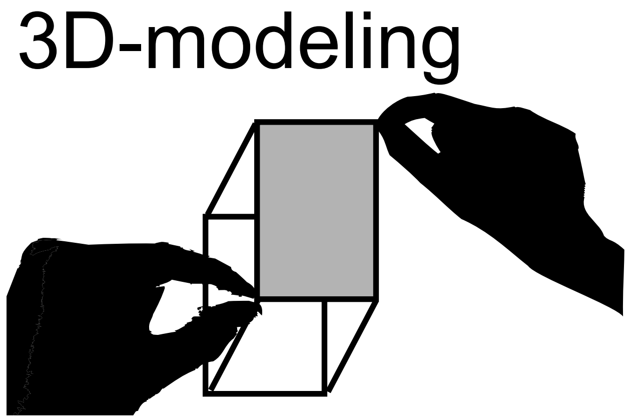

Figure 6: 3D modeling

5. **Sign language translator.** The extension of the finger alphabet detection system is the recognition of the sign language. The sign language uses both hands and considers not only the postures but also the gestures. The need of analyzing the signs in the time domain increases the requirements for the recognition system speed. In the first step the geometrical features of the captured gestures are recognized,

[2]HMD – Head Mounted Display

then the appropriate word is selected from the gesture database using the neural network hardware. The text output can be further processed by the voice synthesis and the system may substitute the human translator.

4 The method of finger detection using correlation

The algorithm for detection of the finger-tips is based on one method used for object classification using the representation by its outline. As it is common in pattern recognition techniques first the objects from sample set are processed to obtain the geometrical description of their shape. Then the features of new, yet not-classified objects are extracted and compared with the teaching set. The object will be classified into the class containing the most similar representer excepting the case of objects with shape too distinct from samples.

As said before, the object outline acts as the input to the classifier[3] and therefore the outline tracing algorithm is used at first. It was implemented according to [8], [9]. The only requirement for outline is the one, that it must be closed. By choosing the classification criterion we have to fulfill the condition of invariance of that criterion according to the translation, rotation and scaling. One approach is to code the boundary by the variation of the direction by the tracing of the contour, then the boundary is represented by the so-called chain-code [12]. The other possibility is to compute the distances of outline points to the center of the object and use these values for further processing [12],[10].

In the implementation of the method the center of the object and the overall length of the contour are calculated at first. The constant number of points (N) from the boundary is selected. For each selected point the distance to the center divided by the boundary length is computed which gives us the discrete function defined on the domain of size N (see figure 7) with values from interval $< 0, 0.5 >$. Lets label such function $D(p)$. This function is invariant according to the translation, rotation and scaling except of shift depending on fact which boundary point is considered as the first one). The right part of the image shows the function corresponding to the boundary of the object to the left. The shape of the outline in the vicinity of the point X is responsible for the peek in the left part of the function graph.

The application of such method for detecting the object features when it has non-constant boundary length needs a minor modification (hand with five visible fingers has almost twice as long contour as the hand with only one visible finger). When detecting a finger we do not need to take into consideration the data from the optionally present carpus and forearm part of the obtained hand image. The global criterion (the boundary is examined as a whole) be-

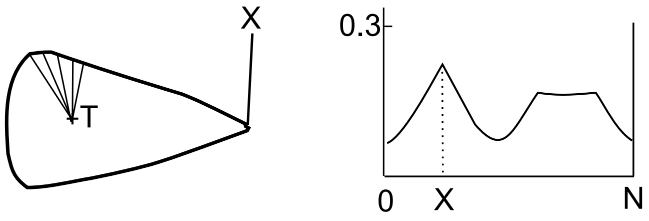

Figure 7: Illustration of the object ond the corresponding function D(p)

comes local (the different parts of the outline are examined separately). The further procedure is equivalent to the algorithm mentioned above. When investigating the boundary segment with N points (N equals to the number of the reference segment points), the length of the segment and the center of it are calculated. Then for each point from segment the function value is calculated by following equation

$$f(P) = \frac{|P - T|}{l}$$

where P is the boundary segment point, T is the center of that segment and l is the segment length.

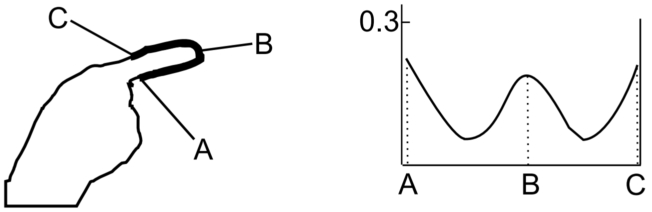

Figure 8: The segment (bold outline part) of the boundary and the corresponding function

After the computation of this discrete function $DF(p)$ defined on the domain of length N the determination of the consistency with the function $Df(p)$ calculated in the same way but for the reference segment takes place. The reference segment is defined interactively by the user (Figure 8 shows the reference segment for the detection of the fingers). The equation for calculation of the difference measure of the examined and the reference segment is

$$Diff(Df, DF) = \sum_{i=1}^{N} | f(i) - F(i) |$$

where $f(p)$ is the distance function for the reference segment, $F(p)$ is the distance function for the examined segment. If the boundary of the investigated object has n points the examination of it requires n-tuple repeating of this step, in the each next step the examined segment is shifted by one point in the boundary array (*modulo n*). The output is now the function defined on the domain of

[3]this is that "cheating" mentioned in the introduction

length n - each value represents the measure of similarity between the reference segment and the corresponding part of the boundary. If the minimum of that function is less than the threshold value for classifying, then we consider that part of the outline as the known object. Figure 8 shows the reference segment for finger detection and its $Df(p)$(function), Figure 9 shows hand gesture and function $Diff(i)$ determining the measure of difference of the gesture hand boundary and the finger outline. The points A and C correspond to the fingers, the part of the outline in the neighborhood of the point B is similar to the finger-root outline.

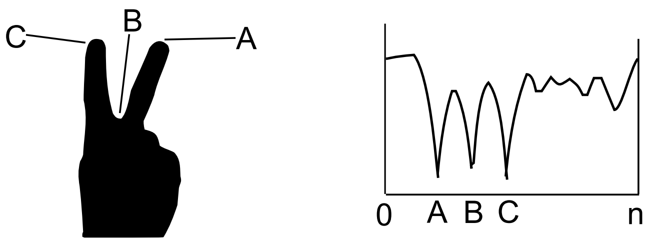

Figure 9: Hand gesture and the function of the difference of the boundary and the finger-tip outline

The complexity of such approach can be enumerated by the equation

$$T(n,N) = (3*N)*n+n$$

where n is the examined object outline length and N is the reference segment length. The time complexity is very important because the need of the real time processing.

The basic steps in the processing:

1. Thresholding of the image and the detection of the hand(s) outline with the removal of the unnecessary objects. The output of this step is the array of the outline points.
2. Calculation of the difference rate for the reference segment and the whole boundary. The output is the discrete function with the values corresponding to the differences of the reference segment and the boundary segments.
3. Selection of the boundary segments similar to the reference (considered as fingers). The selected points from finger-like segments are considered as the finger-tips which are the basic output of image processing part of the system.

Described method can be used for detection of other hands contour features not only the fingers. In the case of occluded objects additional reference objects are needed. The more reference objects are used the more situations can be handled and this is the reason, why we have to pay big attention to the detection methods performance.

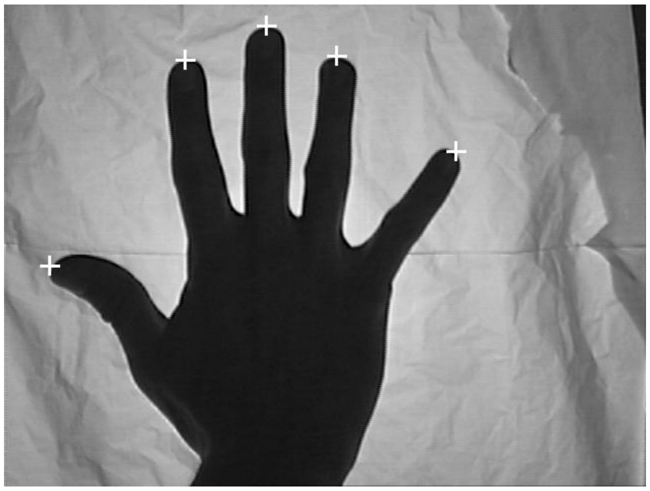

Figure 10: The original image and the output of the method represented by white crosses

5 The method of finger detection using Delaunay triangulation

The triangulation method is an algorithm used for the detection of hands, fingers and finger-tips in the captured frame of the signal from the CCD camera. The algorithm was inspired by the paper [13], but the idea was used for the different goal. Signal passes from the camera to the image-processing stage and after tracing the contours of each object, data are processed by this algorithm.

Some terms used in the description of the algorithm :

- **CDT** – constrained Delaunay triangulation
- **external edge** – an edge of the triangulation, belonging to the border of the object.
- **internal edge** – an edge, completely contained inside of the object, except for endpoints.
- **terminal triangle** – triangle with two external edges
- **sleeve triangle** – triangle with one external edge and two internal edges
- **junction triangle** – triangle with all edges internal
- **chain** – sequence of the midpoints of internal edges, terminating in terminal or junction triangle
- **chord** – a tree consisting of all chains in the triangulation
- **scene** – image captured from the camera

Different types of triangles in the triangulation are shown in the figure 11.

In this figure, terminal triangles are shown in black, junction triangles in white and sleeve triangles are white.

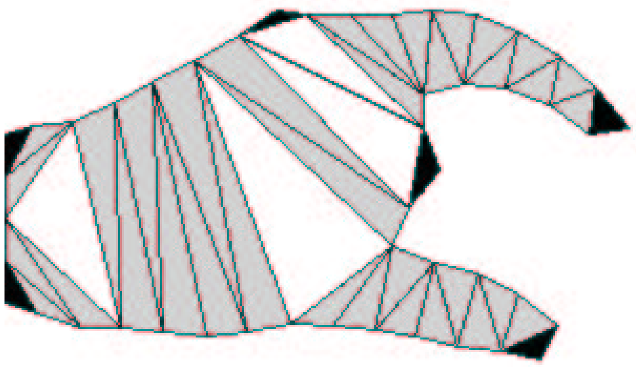

Figure 11: Different types of triangles in the CDT

In the next figure 12, chains and chords are shown in white color. The little black crosses at the ends of the fingers are detected finger-tips.

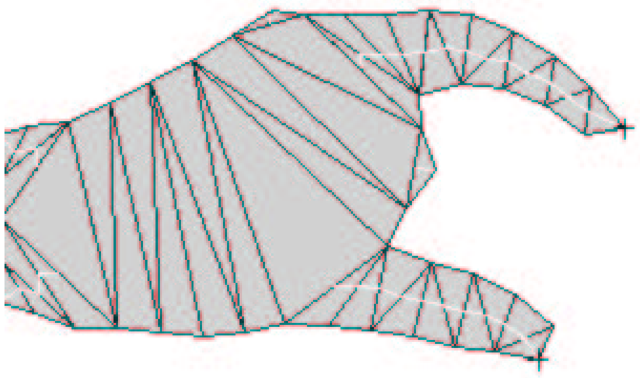

Figure 12: Found chord

Algorithm description :

1. Each separate object in the scene is triangulated using CDT. Triangulation algorithm is set-up in such a way, that it minimizes occurences of triangles with very small angles. For CDT, we use a Triangle package [3].

2. For each object in the scene, the following is done :

   (a) Each triangle in the triangulation is assigned it's type – Terminal, Sleeve or Junction, according to the number of external edges.

   (b) Then the triangle traversal begins. It starts by searching for some Junction triangle, if not found, then for some Terminal triangle. If the triangle mesh is consistent, then there must be at least one. Found triangle is pushed into the stack together with all it's internal edges.

   (c) The following sequence is repeated, till the stack is empty - pop triangle off the stack, if the triangle type is :

      - Terminal, then finish this chain, start a new one.
      - Sleeve, then push a following neighbor to the stack, add this triangle to the current chain.
      - Junction, then finish this chain, start a new one and push both not processed neighbors to the stack.

   (d) Finish the chord data structure. Chains are computed from the triangles by connecting the midpoints of internal edges by the line segments.

3. For each object, the center of gravity is computed as an average of all it's vertices.

4. All chains starting and ending in the Junction triangles are discarded, it is unprobable, that a finger starts and ends in Junction triangles – this is ensured by the optimization of the triangulation because the triangulator is set-up in such a way, that it doesn't create small sharp triangles.

5. All chains exceeding the threshold length are considered to be finger candidates. This threshold was determined empirically, it depends on the relative size of the hand to the size of the scene.

6. For each finger candidate, the reference point is chosen from start and end point as a farthest one from the center of gravity of the current object.

7. If the reference point is far enough from the border of the scene, this finger candidate is considered to be a valid finger and stored into the hand data structure. The reference points are marked as finger-tips. All this can be seen in the figures 12, 13.

This algorithm gives correct results in most cases, but there are weak points. Finger-tip detection is rather heuristics, derived from empirical experience than a geometrically correct algorithm. There are some known cases, when it fails – for example, if the operator moves his fingers too close to the border of the scene. Algorithm is also error-prone, when there is no finger clearly visible. Other disadvantage is it's higher demand on a computational power, compared to pure image-processing algorithms like one described in a section 4.

But this approach has it's advantages too – without farther processing, it is possible to extract far more information from the triangle mesh structure, than from a pure bitmap data. For example when trying to "reconstruct" the 3D model of users hand for displaying or collision testing in 3D. In such case, the inflation algorithm described also in [13] comes handy and this algorithm is based on constrained Delaunay triangulation. This will be probably more used in the future, when we will move to 3D.

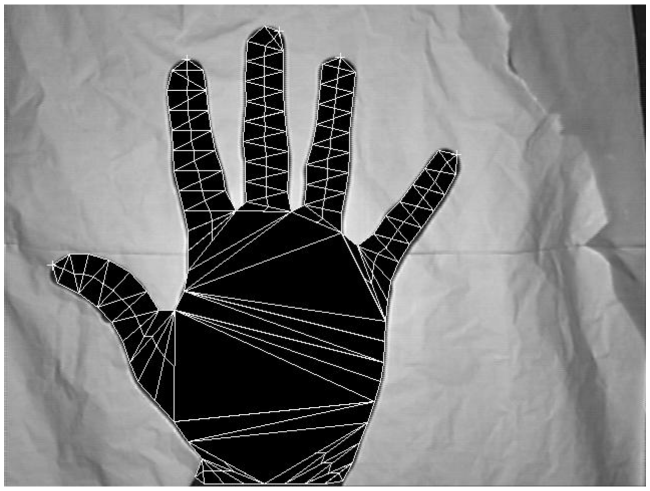

Figure 13: The final result projected on the original signal from the camera

6 Conclusions

This project is work in progress. It still has a long way before producing some practically usable results, but some possibilities seem to be clear already. Interaction with objects using human hands is more straightforward and intuitive, because the operator is doing the same movements as in the non-computer environment.

The two described methods of the finger-tip detection are fast enough to be used for real time processing of the video input. Especially the correlation method is able to achieve frame rates of at least 15–20 frames per second.

Both methods can deal with the artifacts resulting from the image capture and the contour tracing. It is possible to use both of them in parallel, and improve the results of the finger and hand recognition.

Our future work will be focused on the implementation of the applications described in the section 3 and further processing of the results obtained till now. Also there is the need for speeding-up of the programs by the use of the MMX[4] and 3D Now![5] instructions for image processing. We also need to add some basic posture and gesture recognition capabilities to our software. It will be done in the cooperation with the research group of doc. Martin Šperka from the Slovak Technical University. This feature is not implemented yet.

The system is currently developed in parallel on Windows NT 4.0[6] using C++ Builder[7] and on GNU Linux 2.2 using GNU C. In the nearest future, all development effort will be moved onto GNU Linux platform.

Finally, we hope that our operator will not feel like the mentioned dog on the leash anymore :-).

[4]MMX is trademark of Intel corporation
[5]3D Now! is trademark of AMD
[7]Trademark of Inprise

References

[1] http://ls7-www.cs.uni-dortmund.de/bodytalk/. Project BodyTalk homepage.

[2] http://www.cg.tuwien.ac.at/research/vr/studierstube/vt/. Studierstube/Virtual Table homepage.

[3] http://www.cs.cmu.edu/ quake/triangle.html. Triangle Software Package for Delaunay Triangulation.

[4] http://www.cs.cmu.edu/afs/cs/project/vuman/www/boeing/hci.html. Human Computer Interaction Breakout Session.

[5] http://www.deafblind.com/worldsig.html. A – Z to Deafblindness.

[6] http://www.hitl.washington.edu/projects/knowledge-base/virtual-worlds/jove/articles/dsgbjsbb.txt. Recognition of Signals for Combat Formations and Battle Drills.

[7] http://www.novia.ch. Interactive Whiteboards.

[8] Ferko A. and Ružický E. *Počítačová Grafika a Spracovanie Obrazu*. Sapientia, Bratislava, 1995.

[9] B. Beneš, P. Felkel, and J. Žára. *Moderní Počítačová Grafika*. Computer Press, 1998.

[10] Hlaváč V. and Šonka M. *Počítačové Vidění*. Grada, Praha, 1992.

[11] Krueger W. Myron, Gionfriddo T., and Hinrichsen K. Videoplace – an artificial reality. In *Proceedings of the CHI '85 conference on Human factors in computing systems*. San Francisco, USA, 1985.

[12] Gonzales R.C. and Woods R.E. *Digital Image Processing*. Addison-Wesley, 1992.

[13] Igarashi T., Matsuoka S., and Tanaka H. Teddy: A sketching interface for 3d freeform design. In *ACM SIGGRAPH '99 Proceedings*. Los Angeles, August 1999.