Using computer vision in a rehabilitation method of a human hand

J. Katrasnik[1], M. Veber[1] and P. Peer[2]

[1] Faculty of Electrical Engineering, University of Ljubljana, Ljubljana, Slovenia
[2] Faculty of Computer and Information Science, University of Ljubljana, Ljubljana, Slovenia

Abstract— **We developed this program for the purpose of a rehabilitation method that requires a patient to move an object around with his hand. Using a black and white firewire camera the program determines the position and orientation of a black rectangle on a white plane. The user must enter the length and width of the rectangle before the start. With this information the position is determined even if a part of the rectangle is obscured by a user's hand. The program works in real-time (15 to 20 frames per second).**

Keywords— **computer vision**

I. Introduction

One of the major goals of rehabilitation is to make quantitative and qualitative improvements in daily activities in order to improve the quality of independent living. When parts of the brain have been impaired by trauma, incomplete spinal cord injuries and stroke, the functions that those parts of the brain had must be relearned. Relearning is aided by rehabilitation. Relearning is fastest when rehabilitation is done early and if the patient performs task oriented exercises [3]. By using virtual reality in rehabilitation task oriented exercises become more motivating and engaging than formal repetitive therapy. Another positive aspect of virtual reality is that it is programmable, which means that tasks can be adapted to the patient. When the patient advances, tasks can be made more difficult.

Our motivation was to develop a cheap method for measuring position and orientation of an object and use that information in virtual reality exercises. Position and orientation can be measured with commercial products such as OPTOTRAK. The main drawback of using such products is their high price. For example OPTOTRAK costs approximately $150 000. If we could develop a system, that would use only a black and white firewire camera and a PC, this rehabilitation method would be a lot more accessible to the patients, which could then do rehabilitation at home. That would reduce resources needed for rehabilitation and increase the time a patient spends in rehabilitation.

Our goal was to determine if the position and orientation of a black rectangle, with known dimensions, on a white plane could be accurately resolved with a computer vision system in real-time. The system must be able to find out the position of the object even if it is partly obscured with the user's hand.

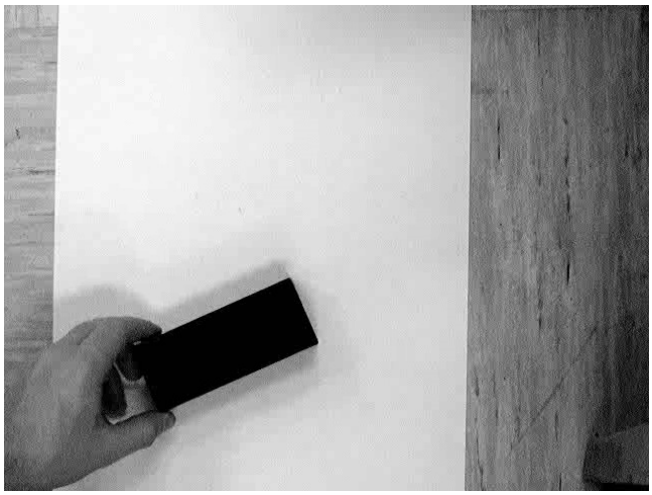

Fig. 1 Captured image

II. Tools and methods

A. Tools used

In developing this system we used some computer vision algorithms already implement in OpenCV [1], an open source computer vision library for C++. We also used this library for capturing images from the camera, displaying images on the screen and saving images to disk. For writing, compiling and debugging the program we used Microsoft Visual Studio 2005. For capturing the scene we used a black and white firewire camera with resolution of 640x480 pixels. The computer used for processing was a PC running Microsoft Windows 2000. The rectangular object was made of wood and painted black.

B. Image processing

We captured the image from camera using OpenCV [1] functions. On the captured image, which can be seen in figure 1, we used the Canny edge detection algorithm, which is implemented in OpenCV [1]. In order to find the rectangle in the picture we first needed to detect straight

lines. We did this with a function cvHoughLines2 with the CV_HOUGH_PROBABILISTIC parameter that returns a sequence of line segments. This function is implemented in OpenCV [1]. These line segments can be seen in figure 3. They are drawn in different colors. Figure 2 shows the output of the Canny algorithm. The Hough transform, Canny edge detection algorithm and their effects are described in [2].

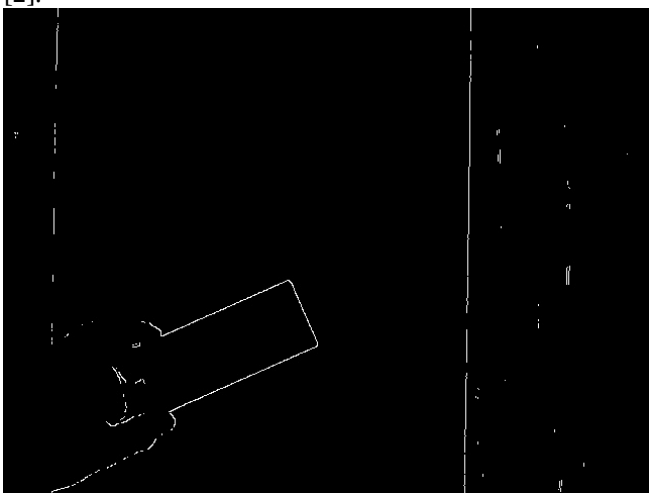

Fig. 2 The result of Canny edge detection algorithm

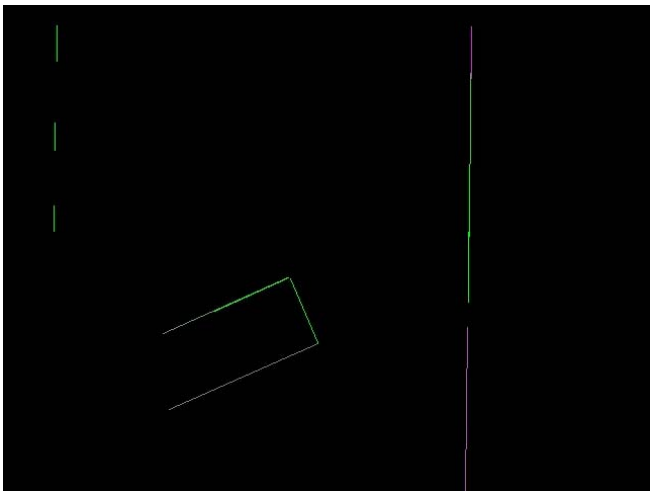

Fig. 3 Straight lines detect with cvHoughLines2

C. Finding the rectangle

When we had the data of all of the line segments in the picture, we needed to analyze that data, in order to find the line segments that form a rectangle. We calculated the angles that the line segments form with the x axis and then we compared these angles with one another. If the difference between two angles was 90 ± 2 degrees, we calculated next parameters:

- the shortest distance between the ends of line segments
- length of both line segments
- the angle between the line segments
- the orientation of the angle that the line segments are forming
- the position of the vertex of this angle

We saved these parameters in a structure representing a right angle. If the orientation and the vertex in multiple angles were very close together we averaged these right angles. We averaged all the parameters, except the lengths of both line segments; instead we kept the longest lengths. The angles, with the shortest distance between the ends of line segments, longer than half of the longest line segment, could not be a part of a rectangle and were therefore eliminated.

If two angles lie on the same line and their orientation is correct, they form a side of a rectangle. So we checked each ray of each angle, if on any of these rays lays a vertex of another angle. If there was another angle we compared the orientations. If the difference in orientations was ±90 degrees the two angles formed a side of a rectangle. Whether this side was the longer or the shorter one, we found out by comparing the length of the line segments. If the line segment lying on the ray, we were checking, was longer than the other line segment of the angle, then that side of the rectangle was the longer one. We only searched for the shorter sides of the rectangle, because the user would probably be touching the longer sides.

One side of the rectangle and the information about the model is enough to calculate the position of the center and the orientation of the rectangle. Dimensions of the model were scaled to fit to the short side found by the algorithm. If two short sides were found we calculated the position and orientation with both of them and then averaged the results.

III. Results

The system calculated the position and orientation of the rectangular object even if someone was holding it with his hand. The system could not detect the object if it was moving to fast, if it wasn't parallel with the plane it was on and if the lighting was inadequate. The system works as fast as 15 to 20 frames per second. The output of the program can be seen in figure 4.

Fig. 4 Rectangle detected by the system

IV. Discussion

The system worked well, if the lighting was good and if the object wasn't moving to fast. This could be improved with higher shutter speeds, which would also affect the sharpness and brightness of the image. This system could be used as a cheaper alternative to OPTOTRAK.

Using this system a simple rehabilitation method could be easily developed. The display of the PC would show a reference object and the object in the patient's hand. The patient would then have to move the object he is holding in the position indicated by the reference object. The reference object would move around the screen and the patient would have to follow it. The progress of the patient would be measured by calculating the mean square distance between the objects and the mean square difference in orientations of the objects in a certain amount of time. The faster the reference object would move around the screen the more difficult the exercise would be.

The system would be a lot more useful if it worked in three dimensions. This application indicates that determining the position and orientation of a rectangular box in three dimensions could be done with algorithms similar to the ones used in our program. Each face of the rectangular box would have to be in different color and a color camera would be necessary.

V. References

1. OpenCV library at http://sourceforge.net/projects/opencvlibrary/
2. Russ J C (1995) The Image Processing Handbook. Boca Raton
3. Sveistrup H (2004) Motor rehabilitation using virtual reality. Journal of NeuroEngineering and Rehabilitation 1:10

Author address:

Author: Jaka Katrasnik
Institute: University of Ljubljana, Faculty of Electrical Engineering
Street: Trzaska cesta 25
City: SI-1000 Ljubljana
Country: Slovenia
Email: jaka.katrasnik@gmail.com