PERIODICUM BIOLOGORUM
VOL. 115, No 4, 535–544, 2013

UDC 57:61
CODEN PDBIAD
ISSN 0031-5362

Original scientific paper

A computer vision based system for a rehabilitation of a human hand

PETER PEER
ALEŠ JAKLIČ
LUKA ŠAJN

University of Ljubljana
Computer Vision Laboratory,
Faculty of Computer and
Information Science
Tržaška 25, SI-1001 Ljubljana, Slovenia

Correspondence:
Luka Šajn
University of Ljubljana
Computer Vision Laboratory, Faculty of
Computer and Information Science
Tržaška 25, SI-1001 Ljubljana, Slovenia
E-mail: luka.sajn@fri.uni-lj.si

Key words: computer vision,
3D reconstruction, cuboid model, real time
execution, web camera, arm rehabilitation,
injury, stroke

Received September 24, 2013.

Abstract

Paper presents a rehabilitation system for patients who suffer from arm or wrist injury or similar. The idea of the rehabilitation using computer and additional hardware is not new, but our solution differs significantly. We tried to make it easily accessible and thus started with a limitation that only a personal computer and one standard web camera is required. Patient holds a simple object, cuboid, and moves it around. Camera records his movement while the software in real-time calculates position of the object in 3D space on the basis of color information and cuboid model. Object is then placed in the virtual 3D space, where another similar object is already present. The patient's task is to move the real object in the position, which matches the position of the virtual object. Doing so the patient trains specific movements that speed up the recovery. Evaluation of the system shows that presented solution is suitable in cases where accuracy is not very critical and smaller 3D reconstruction deviations do not thwart the process of rehabilitation.

INTRODUCTION

According to the World Health Organization, 15 million people have a stroke worldwide each year. Of all people who have a stroke, about a third are likely to die within the first ten days, about a third are likely to make a recovery within one month, and about a third are likely to be left with disabilities and needing rehabilitation. The types and degrees of disability that follow a stroke depend upon which area of the brain is affected. Generally, stroke can cause five types of disabilities: paralysis or problems controlling movement; sensory disturbances including pain; problems using or understanding language; problems with thinking and memory; and emotional disturbances. The paralysis is one of the most common disabilities resulting from stroke. The paralysis is usually on the side of the body that is opposite to the side of the brain affected by stroke: it may affect the face, an arm, a leg, or the entire side of the body. Movement impairments after stroke are typically treated with intensive, hands-on physical and occupational therapy for several weeks. Unfortunately, due to economic pressures on health care providers, stroke patients are receiving less therapy and going home sooner *(1–3)*. Additionally, there are various types of physical injuries, for instance due to car or sport accidents, where rehabilitation is required. Therefore we developed a system, called FRI Rehab 3D, which would help the patients to continue the rehabilitation and practice of intensive movement training also at home without the expense of an always-present therapist and special expensive equipment.

We explored existing solutions in Section 2, which are based on computers, robots, sensors and cameras. Since most of these systems require specialized hardware they can be very expensive, not always easy to use, and difficult to carry or set up at home. Thus, the biggest differences between existing systems and our solution are the price and the availability of required hardware. Whereas some solutions can cost from few thousand USD (not counting in a personal computer) up to over one hundred thousand USD, the cost of our solution is merely a personal computer and a single web camera, which people usually already have. Our goal was therefore to develop a solution that would allow rehabilitation for as many people as possible.

We developed and tested our system on a personal computer with Dual Core 1.86 GHz processor and a Logitech Quickcam Pro 5000 color web camera. The system was running Windows XP and the software was developed in Microsoft Visual Studio 2005 in C++ programming language. We decided for Windows family operating system because it is by far the most widely used operating system and thus probably preferred choice for most users *(4)*. We extensively used an open source package OpenCV, which offers many useful algorithms to solve computer vision and graphics related problems *(5–8)*.

In Section 3 we describe how to set the working space environment to achieve best results. Discussion of segmentation methods can be found in Section 4. Next, the step of moving into 3D space based on the information from segmentation process is described in Section 5. In Section 6 the process of matching the real and virtual object is presented. Results of the quantitative and qualitative experiments are presented in Section 7 to demonstrate effectiveness of the proposed solution. Finally, Section 8 gives an overall conclusion and presents possible directions for improving the functionality of our system.

EXISTING SOLUTIONS

The first examined solution, Rehab 2D, was also developed at the University of Ljubljana in Slovenia *(9)*. This method requires only a personal computer and a monochromatic camera, which tracks and detects the object that is moved by the patient. In the environment with sufficient lightning and under the condition that the object is not being moved too fast, the system successfully detects the object even if it is partly occluded. Unfortunately this method works only in two dimensions, so we expanded the basic idea of this solution and moved into three dimensions by using color information and cuboid model.

The MTi is a miniature size and low weight 3DOF Attitude and Heading Reference System (AHRS). The MTi contains accelerometers, gyroscopes, and magnetometers in 3D. Its internal low-power signal processor provides real-time and drift-free 3D orientation as well as calibrated 3D acceleration, 3D rate of a turn and 3D earth-magnetic field data. As such it could also be used for rehabilitation purposes. The price of the system is about 3,400 USD. According to the specifications the error of a moving sensor is less than two degrees [10]. This means that for the cuboid that we used for testing our system (length 14 cm, width 8 cm, height 6 cm), the error would be less than 5.35 mm.

The system Optotrak Certus provides accuracy of up to 0.1 mm and resolution of 0.01 mm. It can track up to 512 markers, with a maximum marker frequency of 4,600 Hz. Their motion path is tracked in 3D by the three infrared cameras. This system costs about 150,000 USD. It is not limited though only to rehabilitation, but is used in different industries, biomechanics, universities, and research institutions around the world *(11–12)*.

Y. Tao and H. Hu developed a real-time hybrid solution to 3D arm motion tracking for home-based rehabilitation by combining visual and inertial sensors. The Extended Kalman Filter (EKF) *(13)* and Particle Filters (PF) *(14)* were used to fuse the different data modalities from two sensors and exploit complementary sensor characteristics. Due to the non-linear property of the arm motion tracking, upper limb geometry information and the pin-hole camera model are used to improve the tracking performance. With additional optimization algorithms and introduction of additional constraints they achieved the accuracy of ± 5 cm.

ARMin is a system for robot-aided arm therapy *(15)*. A pilot study with ten healthy subjects and six patients was carried out to analyze comfort, functionality and acceptance of the ARMin training. Using a scale from 1 to 10 an average mark of 8.5 was assigned by the patients. In a second pilot study, the effects of the ARMin training were analyzed with three chronic stroke patients. After eight weeks of training there was a reduction in motor impairment assessed by the upper limb portion of the Fugl-Meyer Assessment. The improvements in this assessment are in a similar range as in other studies on arm therapy in chronic stroke patients. Unfortunately, quantitative results are not presented.

Here we can also mention the Stroke Rehabilitation Exerciser from Philips Research, which uses wireless motion sensors to fully capture the patient's upper-body movements. The idea is to develop a motion sensor exerciser to allow patients to perform vital rehabilitation exercises at home. This system is currently in the prototype state and is being tested and refined in clinical settings *(16)*.

PREPARATIONS

One of the major factors that affects the efficiency of our solution, FRI Rehab 3D, is the work space arrangement (Fig. 1). First, we have to understand that the only thing that is important on the picture that the camera captures is the cuboid. Therefore we have to make sure that it is easy to segment it out from the whole picture. We use the technique called blue or green screen, which is commonly used in weather forecast broadcasts, but with black color *(17–18)*. Therefore we use a black cur-

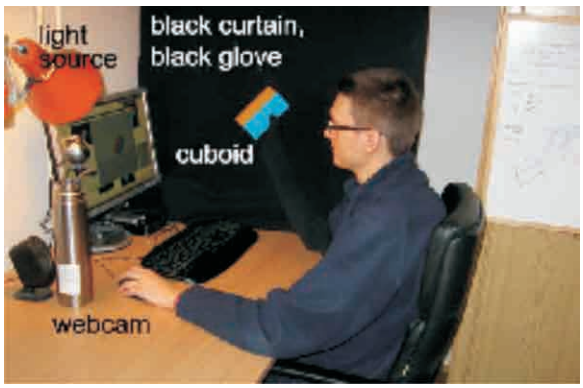

Figure 1. *Work-space arrangement.*

tain for the background and wear a black glove on the hand with which we hold the object. This makes it easy to segment out the cuboid.

Since we are limited to web cameras, which are quite sensitive to lightning conditions, we have to pay special attention to the light sources in the room. Ideally the room should be well lit and the cuboid should be illuminated evenly from all sides. We noticed that the combination of daylight and one or two artificial lights work very well. The lights should not be pointing directly towards the cuboid but should produce more diffuse light.

The material, that the sides of the cuboid are made of, plays a big role. The material should be bright enough, but not too reflective. We used a wooden cuboid, which sides we precisely covered with colored matt photograph paper. Our cuboid reconstruction method is based on detection of cuboid corners, thus appropriate uniform colors of the sides are the easiest and fastest way to obtain the edges and corners, i.e. the cuboid framework. The three needed colors are specified in the HSV color space. In this color space the definition of three as different as possible colors is quite straightforward: the interval in which the parameter H is defined has to be split into the three equal parts. Knowing that, the colors of the sides were defined as: (H=60, S=100, V=100), (H=180, S=100, V=100), (H=300, S=100, V=100).

Another important aspect of the cuboid is its size which has to be input before the start of the rehabilitation process. On the basis of cuboid dimensions and its colors the algorithm can construct a correct cuboid model and use it for 3D reconstruction.

In the preparation stage the camera's parameters, such as brightness, contrast, white balance, exposure, and gain, should also be configured properly. Usually the camera's driver sets these parameters automatically, but in our experience it was always better to set and adjust them manually. We also need to know the camera's field of view and its resolution to compute 3D coordinates from the information obtained from the original, captured image *(19, 20)*.

SEGMENTATION

Since our approach does not use specialized external hardware, the computer processor has to do much more work and therefore our method can be computationally very demanding. Consequently it is necessary to implement time-efficient algorithms. Thus, in the first step we crop the image to get only the region of interest (using cvBoundingRect *(5)*).

As will be described in more detail in the following sections, we represent the cuboid in the virtual 3D space with eight corners that are computed from detected corners in the input image. Therefore, it is important to develop a good method for identifying the corners in the captured image. Fig. 2 gives basic steps of the whole system in a flow-chart.

The whole problem of identifying the corners is split into three subproblems: if one side of the cuboid is visible, if two sides are visible, and if three sides are visible. The first step is to determine how many sides are visible – one, two or three. This is done using the histogram of H values on the image converted to HSV color space *(21)*. Then the H interval is split into bins according to the colors of the cuboid sides, which were input in the

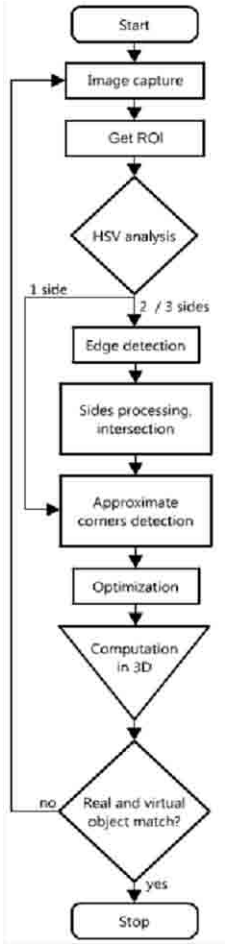

Figure 2. *Basic steps of the whole system.*

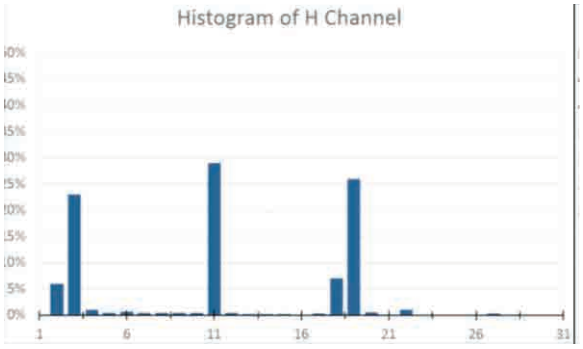

Figure 3. *Histogram of H values of an image, where three cuboid sides are visible.*

preparation stage. Every pixel of the picture is then put in proper bin on the basis of its H value. In our case this method gave very good results. To reduce the calculation time the picture is resized to typically one eighth of its original size. This speeds up the process but leaves the ratios between bins practically the same. Ratios between bins determine which sides are present on the image and consequently which of the three subproblems needs to be solved. The histogram of H values in the case where three sides are visible is shown in Fig. 3.

One side

This subproblem is solved in two steps. In the first step we detect approximate corners, which are then optimized in the second step (Fig. 4). The approximate corners are detected by finding the smallest bounding rectangle around the visible cuboids' side. The bounding rectangle touches the cuboid side at four points, which are approximate corners. These corners are further refined with local optimization in area close to the corner. For every white pixel in this area the number of white surrounding pixels is calculated. The pixel with the least white surrounding pixels is most likely the corner.

Two sides

In the case when two sides are visible the intersection method is used. Here the edges of the cuboid are detected first. The original image is split on the basis of color information in two images with only one cuboid side per image. Each image is further processed to remove any non-belonging surface and dilated to make it a bit bigger and more smooth. Then the intersection of both images is performed, which leaves only a narrow line of pixels that represents the edge. This set of pixels is further narrowed, eroded. On this basis the line defining the edge is calculated with linear regression. Fig. 5 shows detection of one line defining the edge on the example where three sides are visible.

From the line defining the edge start and end points of the edge can be obtained. The line is traced until non-black pixel is found. This is the start of the edge. Then the line is traced further until a black pixel is detected.

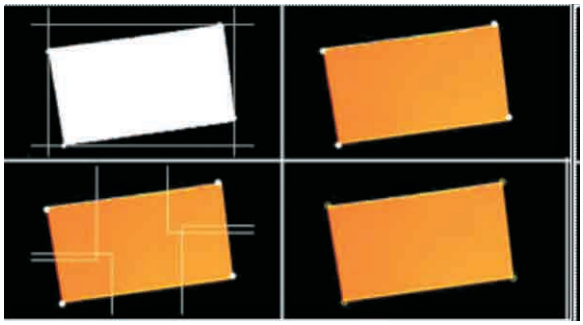

Figure 4. *Upper two figures show approximate corners, which are optimized with local optimization visible in two lower figures.*

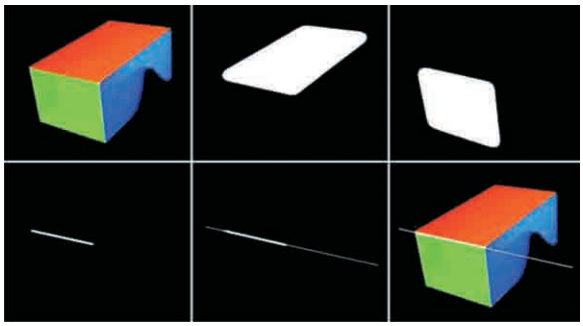

Figure 5. *Obtaining the edge information by intersecting two cuboid sides after isolating and processing them.*

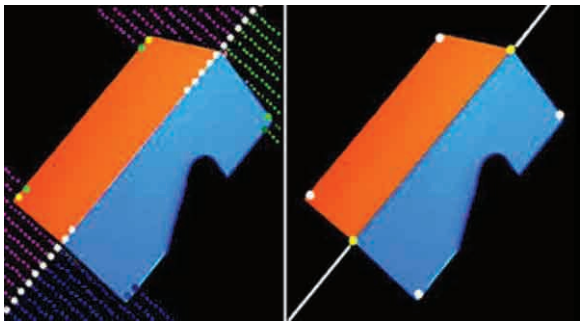

Figure 6. *Detecting the outer four corners by analyzing perpendiculars along the edge.*

This is the end point of the edge. Thus, the first two corners are detected.

We continue by detecting the remaining four corners. In the vicinity of the previously detected corner (beginning of the edge) *n* perpendiculars on the edge are defined. Each perpendicular starts at the edge and is tracked till the outer edge of the cuboid. If all pixels of a perpendicular are black, its length is set to zero. If the perpendicular runs across the cuboid, its length is calculated. The new corner is found when the algorithm detects two consecutive perpendiculars with big length difference (while both perpendiculars in the next pair are of the maximal length found in the vicinity of the already found

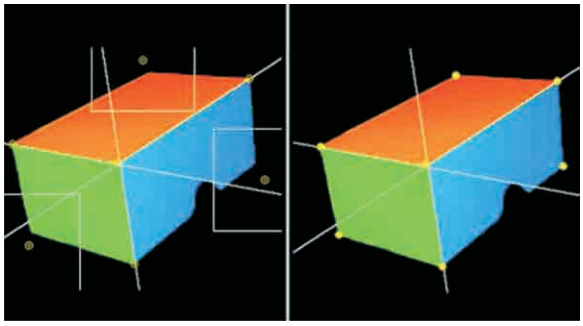

Figure 7. *Left figure shows the first four corners and three approximately detected corners. The figure on the right shows the result after local optimization.*

corner). This new corner is further refined by the already mentioned local optimization. This method is repeated also for the other three remaining corners. Fig. 6 illustrates this process.

Three sides

The first step of this subproblem is similar to the subproblem of two cuboid sides. Here though sides intersection method is performed three times for the three pairs of sides and so three edges are detected. Intersection of all three edges defines one of the corners of the cuboid, let us call it the central corner.

Next three corners are detected by following the three edges. For each of these three corners we start at the central corner and trace the edge until a black pixel is found. Additional check based on color information along the edge is performed, which confirms that the resulting corners are the right ones; note that the line defining the edge can be traced in two directions.

Now, there are only three unknown remaining corners. These can be obtained by mirroring the central corner across the middle of the lines defined by the pairs of in the previous step detected three corners. After local optimization is applied, these three corners are usually well detected (Fig. 7).

MOVING INTO 3D SPACE

The 3D coordinates are calculated when all the characteristics, namely sides colors, number of sides and corners are known. To calculate all eight corners of the cuboid it is theoretically enough if we detect only four corners of one side. But due to the fact that the detection is not always very precise, we get better result by including other corners as well. This is possible of course only when two or three sides are present. The method for obtaining 3D coordinates is therefore not completely the same for the three subproblems, though they are all based on the same idea. To obtain 3D coordinates the following data is required:

- the 2D coordinates of corners,
- the cuboid dimensions,
- the camera resolution,
- the camera field of view or focal length.

The detection of the corners and their 2D coordinates was already described in the previous section. In addition, it is necessary to know which side of the cuboid each detected corner belongs to, so the distances between the corners can be identified. The resolution of the camera can be set directly in the software, though we have to be careful that the camera supports the resolution that we set. We could not find the specification of camera's field of view or focal length so we had to measure it ourselves.

Figure 8 shows how an object is projected on the image plane. In this case the object is just one side of the cuboid. The camera is put in the center of the 3D coordinate system and is marked with label o. Points on the image are labeled a, b, c and d. These represent the corners that are detected on the captured image. Labels a', b', c' and d' refer to the actual corners of the cuboids' side that we need. Since focal length f is the distance between the camera and the image plane, we can extend the 2D corners a, b, c and d in the following way:

$$(x, y) \rightarrow (x, y, f) \tag{1}$$

Thus, the new values of **a**, **b**, **c** and d are:

$$\begin{aligned} a &= (x_a, y_a, f) \\ b &= (x_b, y_b, f) \\ c &= (x_c, y_c, f) \\ d &= (x_d, y_d, f) \end{aligned} \tag{2}$$

We introduce four new scalars k_1, k_2, k_3 and k_4:

$$\begin{aligned} \overrightarrow{oa'} &= k_1 \cdot \overrightarrow{oa} \\ \overrightarrow{ob'} &= k_1 \cdot \overrightarrow{ob} \\ \overrightarrow{oc'} &= k_1 \cdot \overrightarrow{oc'} \\ \overrightarrow{od'} &= k_1 \cdot \overrightarrow{od} \end{aligned} \tag{3}$$

We can see that the scalars k_1, k_2, k_3 and k_4 are uniquely defined if two limitations, which are based on the cuboid model, are introduced. First limitation takes into consideration the intersection of both diagonals of the cuboid side. The first diagonal is defined by a' and c', the second diagonal is defined by b' and d':

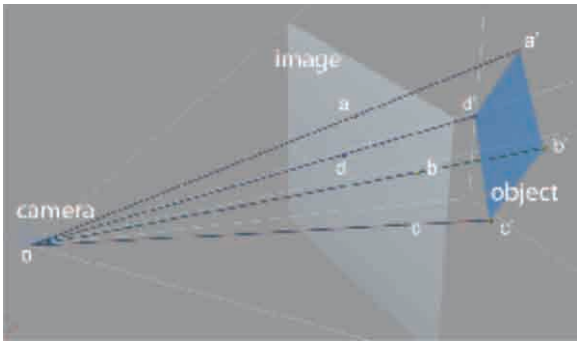

Figure 8. *Detailed illustration of how an object is projected on the image plane.*

$$a' \cdot c' = b' \cdot d' \quad (4)$$

The second limitation is the length $l(a',b')$. Note that the lengths are given in the preparation step (Section 3), where we input the dimensions of the cuboid.

Both limitations together form four equations, which uniquely define needed four scalars k_1, k_2, k_3 and k_4:

$$\begin{aligned} x_a \cdot k_1 + x_c \cdot k_3 &= x_b \cdot k_2 + x_d \cdot k_4 \\ y_a \cdot k_1 + y_c \cdot k_3 &= y_b \cdot k_2 + y_d \cdot k_4 \\ z_a \cdot k_1 + z_c \cdot k_3 &= z_b \cdot k_2 + z_d \cdot k_4 \end{aligned} \quad (5)$$

Thus, the scalars k_1, k_2, k_3 and k_4 are calculated and real cuboid corners a', b', c' and d' are obtained from Eq. (3).:

$$\begin{aligned} a' &= (x'_a, y'_a, z'_a) \\ b' &= (x'_b, y'_b, z'_b) \\ c' &= (x'_c, y'_c, z'_c) \\ d' &= (x'_d, y'_d, z'_d) \end{aligned} \quad (6)$$

This concludes the calculation of the 3D coordinates of the four corners of the first side. The remaining four corners are calculated using normalized vector product. With altogether eight corners we can fully reconstruct the cuboid and visualize it in the virtual 3D space.

MATCHING

After obtaining 3D coordinates the cuboid is placed in the virtual 3D space in which another cuboid with the same characteristics as the real one is placed at for therapy appropriate position and orientation. The goal of the user is to move the real cuboid and to match it to the virtual cuboid as well as possible. In this way the user trains specific movements of his hand that assist in the recovery after the injury.

Detection of the matching of the real and virtual cuboid is a two step process. In the first step the centers of both cuboids are calculated. If the two centers are close enough, we move on to the next step, where distances between corresponding corners of real and virtual cuboid are calculated. When these distances are under a predefined limit (defined by the therapist), matching is detected and a success message is displayed (Fig. 9). The parameters that define minimal distance can be set in the initialization file of the system. Setting these two parameters to higher or lower value the difficulty can be adjusted to fit the user's needs.

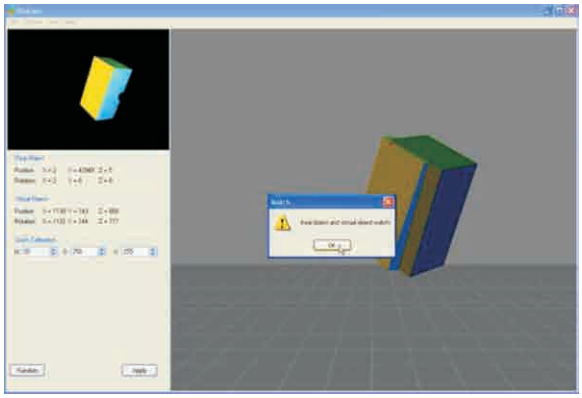

Figure 9. *Successful matching of the virtual and real cuboid displays a success message. (Live image is on the left.)*

RESULTS

The system was developed and tested on a personal computer with the Intel Core2 Duo 1.86 GHz processor. Logitech Quickcam Pro 5000 web camera was used to capture images at resolution of 640×480 pixels. On such equipment the achieved frame rate of our system was usually somewhere between four and nine frames per second. This is not a lot, but we have to keep in mind that the used equipment was not very fast and that our system at this point has not been optimized with regards to the computing time.

Furthermore, since lens distortion is not corrected on images, the reconstructions inherit its error. Processing undistorted images in general brings better reconstruction results, but in our solution processing distorted images does not give the impression of any worse performance to the end user, which is of main importance in our solution. Having in mind that undistorting image sequence means that much more processing time is needed, we should be satisfied with the results gained using the distorted sequence if we want to run in real-time. Another drawback of undistorted images is that they are more blurred in comparison to distorted originals, which can influence the subsequent processing steps *(22)*.

Two types of tests were performed: a quantitative and a qualitative analysis. But before we describe both of them, let us take a look at commonly occurring problems.

Common problems

In some cases the corners were very inaccurately detected due to the environment conditions. One of the common reasons was the bad lightning in the room (Fig. 10). When there is not enough light the sides are not completely visible and the result can be quite unpredictable. Fortunately, it is easy to add more light sources and avoid this problem. But when adding more light, we have to be careful not to cause another possible problem. If the light source is too strong and directed towards the object, a very strong reflection can occur on the object, making the side appear to be of different color (usually white or very bright) than it actually is (Fig. 11). Again, the result is incorrect. This can be avoided by careful setting of the light and by using non reflective material for the cuboid sides. The third problem occurs when the object is moved too far away from the camera. Here not much can be done, except using higher resolution. This though is not a very good solution since it requires much more computational time. However, if the maximal distance is not more than 85–90 cm (which proved to be more than acceptable in practice), the accuracy and stability is good enough.

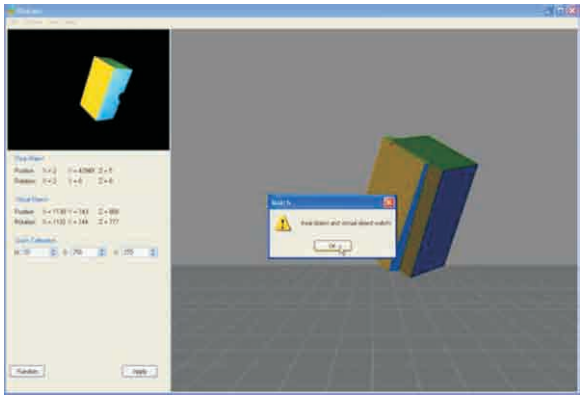

Figure 9. *Successful matching of the virtual and real cuboid displays a success message. (Live image is on the left.)*

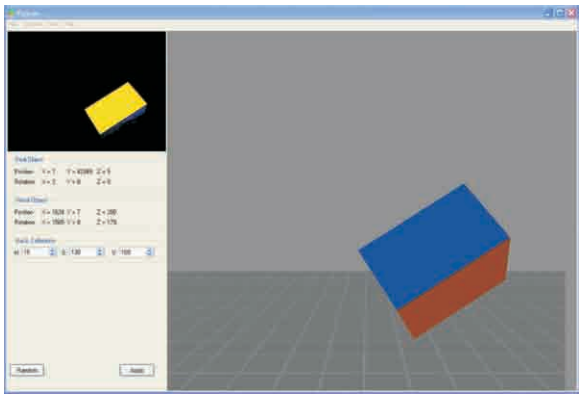

Figure 10. *Inaccurate reconstruction due to bad lightning.*

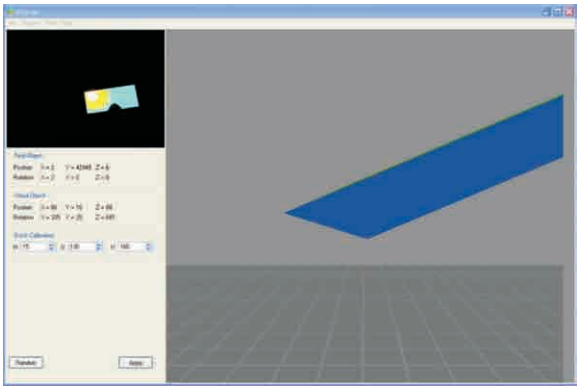

Figure 11. *Strong light source is directed towards the object, causing a very strong reflection on the object, and thus inaccurate reconstruction.*

Quantitative analysis

Proper evaluation normally demands that we test the proposed solution on synthetic and real scenes, to perform quantitative and qualitative evaluation. As it is very hard (if not even near to impossible, especially without a big budget to buy a device like Optotrak *(11, 12)* or at least MTi *(10)* to get ground truth information for real scenes of appropriate quality, but we can assure perfect ground truth information for synthetic scenes, we choose synthetic scenes for quantitative evaluation.

In the case of Tao and Hu system *(14)*, which is the closest to our system in comparison to the cost and employed sensors, the tracking results from the marker-based system Qualisys (not their system) are regarded as a ground truth. But the error of Qualisys system itself is not reported and taken into account in their evaluation. Furthermore, they perform the evaluation on simple circular motion on the desk, which actually is a 2D example.

Some of the real world conditions aren't taken into account in the synthetic scenes, but the most obvious are. All the real world conditions are taken into account in our qualitative evaluation, where the user is the one making the judgment, similar as in the ARMin system *(15)* evaluation.

The quantitative test was performed mainly to identify the ideal capabilities of the system and to evaluate the main idea of the 3D reconstruction using single camera, color information and cuboid model. To get the perfect conditions, i.e. the ground truth information, we used 3ds Max application and in it we defined a space very similar to our own workspace arrangement. We placed a camera, light sources and a proper cuboid in this virtual space. We rendered the scenes at the resolution of 640×480 pixels and used rendered images as an input to our own system. In the end we compared the coordinates of real corners (of 3ds Max scene) and calculated corners obtained by our solution. Two types of errors were defined in this experiment. A quantitative error was defined as a distance between coordinates of corner in 3ds Max and a calculated corner. Since this type of error does not always give a correct feedback about how the error appears to the user, second error estimator was defined, which we call approximate qualitative error. This type of error neglects the error on z axis, because the error on this coordinate is much less noticeable to the user than the errors on x and y axes. For both types of errors several metrices were calculated: average error with standard deviation, geometric mean and median *(23, 24)*.

Altogether 45 cases with 226 corners (15 cases with one visible side, 15 cases with two visible sides and 15 cases with three visible sides) were examined. The results are shown in Tables 1 and 2.

Figure 12 shows the correlation between the distance of the object from the origin of the coordinate system (camera) and error size for both types of errors. Both graphs are also modelled with linear functions which, as we would expect, show that the error increases with the distance.

Figure 13 shows the distribution of the error sizes, where we can notice the difference between the quantitative and the approximate qualitative errors functions. The approximate qualitative errors function has much

TABLE 1

Calculated errors on the basis of 45 cases and 226 corners.

	geometric mean	median	average
quantitative	11.73 mm	11.58 mm	19.38 mm
approx. qual.	2.51 mm	2.34 mm	4.42 mm

TABLE 2

Average quantitative error expressed in the percentages of the actual distances.

	average error ± standard deviation
distance from origin	3.02 % ± 3.84 %

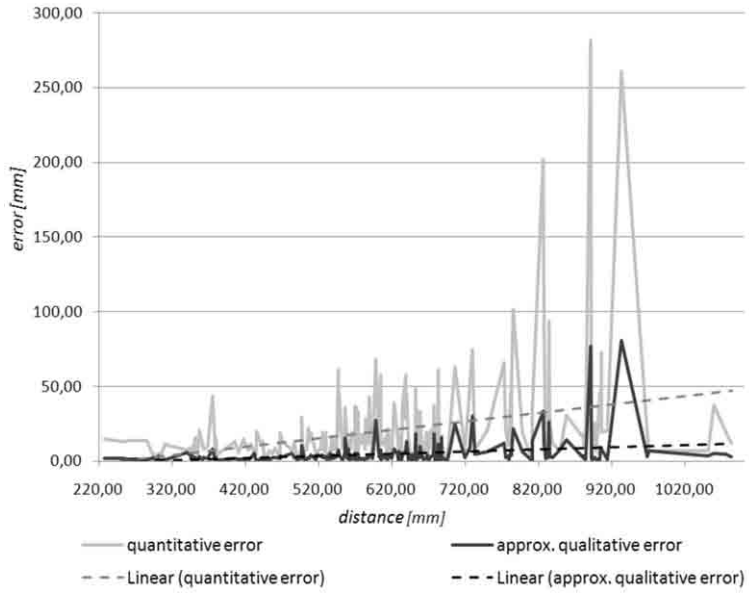

Figure 12. *Graph shows the correlation between the distance from the origin and the error size.*

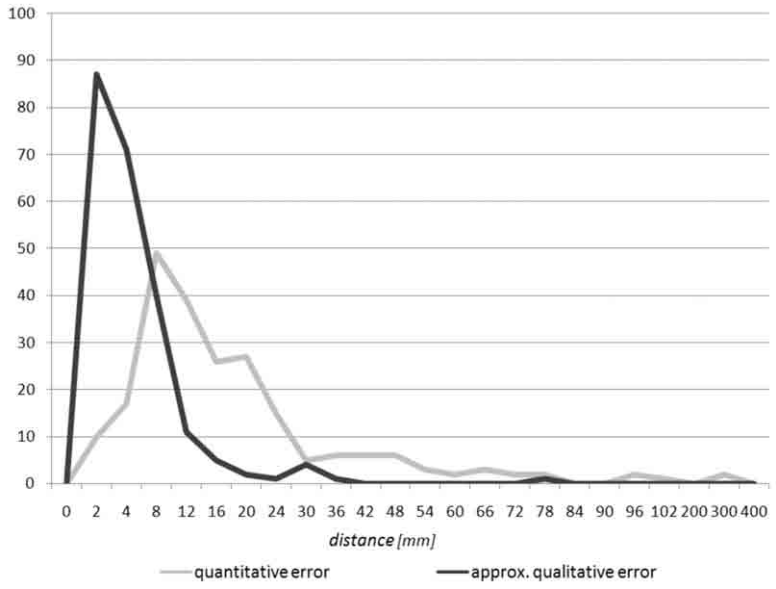

Figure 13. *Distribution of the quantitative and the approximately qualitative errors.*

narrower area of the expected errors than the quantitative errors function, which means that the stability from the user point of view is better than what one would expect looking only at the quantitative errors function.

Table 3 compares the quantitative errors of other methods with our solution. We can see that the more expensive methods with specialized hardware are more accurate, but their cost/error ratio is not so good.

TABLE 3

Comparison of different solutions (see Section 2 for more details).

	reported error	cost
FRI Rehab 3D	11.58 mm	price of the camera
Optotrak [11, 12]	0.1 mm	approx. 150,000 USD
Xsens MTi [10]	max. 5.35 mm	approx. 3,400 USD
Tao, Hu [13, 14]	50 mm	price of the camera and the sensor

Qualitative analysis

The second test was the qualitative analysis of our system which was performed on real input from the camera. Before we initiated the test, we calibrated the colors again and re-adjusted the light sources to get the best conditions and therefore best results. The test comprised of making three videos and evaluating each frame. In the first video (2160 frames, 72 seconds) we moved and rotated the cuboid so that most of the time only one side was visible at once. In the second video (2677 frames, 89 seconds) we moved and rotated the cuboid so that mostly two sides were visible, and in the third video (3477 frames, 116 seconds) we moved the cuboid so that three sides were visible at once. Then, we introduced three grades: good (no deformation), acceptable (the cuboid is in the right position, but one of its sides is only a bit deformed), and bad (very deformed cuboid). According to them we graded each frame of all three videos. Table 4 summarizes the results.

The results show that of all frames only 4.2 % were incorrect, i.e. graded acceptable or bad. However, only 1.2 % are bad and other 3 % are acceptable. The results are very satisfactory, especially if we take into consideration that the algorithms used can be further optimized, as it will be described in the next section. These results are possible though only with proper lightning adjustment, correct color calibration, good camera settings and can vary from video to video due to different factors (the distance of the cuboid from the camera, the speed of movement, the quality of the cuboid etc.).

TABLE 4

Qualitative analysis results.

video	good	acceptable	bad
#1 [frames]	2107	0	53
#1 [percentage]	97.5 %	0 %	2.5 %
#2 [frames]	2553	93	31
#2 [percentage]	94.5 %	4.3 %	1.2 %
#3 [frames]	3307	154	16
#3 [percentage]	95.1 %	4.4 %	0.5 %
all [frames]	8314	247	100
all [percentage]	95.8 %	3.0 %	1.2 %

CONCLUSION

Our system, FRI Rehab 3D, uses very different approach than other, more expensive systems. Whereas they use advanced and specific hardware, we limited the requirements to only a single web camera. This brings both advantages and disadvantages:

Main advantages:

- low price and affordable hardware,
- can be used even at home,
- accuracy will be better with optimized algorithms and better cameras.

Main disadvantages:

- accuracy is lower than in more expensive systems,
- sensitivity to lightning conditions,
- computationally demanding.

Basically, we can separate our method in two steps. In the first step as many characteristics as possible are found in the captured image. The results suggest that this step works quite good, though it would be possible to improve it by additional analysis of the most commonly occurring errors. The second step calculates 3D coordinates from previously detected characteristics, by using color information and the cuboid model. This step is theoretically easily and uniquely solvable, but as it turns out in practice, it is not always very accurate. One of the main reasons is low camera resolution. This problem could be avoided by using better camera with higher resolution and with more time efficient algorithms. And, indeed, the errors of tests at resolution of 1,600×1,200 pixels were typically two or three times smaller (but so was the frame rate). Fortunately, using higher resolution is not the only possibility to improve the accuracy and the stability of detection and usability of the whole system. Here we list a few more ideas:

- The algorithms could be optimized to reduce computational demands.
- The accuracy of the 3D reconstruction could be significantly improved by implementing an algorithm that would use the cuboid model in a more efficient way. For instance, if a correction step of all corners together is added after detecting eight corners according to the constraints on the whole cuboid, the precision could be improved.
- One of the possible improvements comes from the observation that even a small change in 2D coordinates of a corner can cause big changes in 3D reconstruction. This is especially noticeable at bigger distances. If, for example, at one moment the corner is well detected, but at the next moment it is off by just a pixel, the error can increase significantly. Therefore, we need subpixel accuracy *(25,*

26). This could be done in the following way: first, the 2D coordinates would be detected in the same way as they already are. Then two neighboring sides would be calculated. Two corners of these two sides that touch would be compared and distances between them calculated. Ideally, the distances would be as low as possible. Then the 2D coordinates would be slightly changed and two sides calculated again and distances between the corners re-evaluated. This process could be repeated until the error is small enough, although the number of iterations would have to be limited to retain real-time execution.

- The system could compare new detection with previous one and check if the change is logical. If it is not, it could re-evaluate or skip the frame.
- The reconstruction results could be improved by eliminating the lens distortion problem, i.e. by undistorting the images in the preprocessing step *(22)*.
- The stereo reconstruction approach could be implemented *(27, 28)*.
- The system could also include patients database and their statistics, which could be analyzed by physiotherapist to evaluate the improvement of the patient.

Based on presented results and discussion, we conclude that under the condition that the user is willing to put in some effort on his own, the presented system serves its purpose good enough despite some faults. The presented system is suitable mostly in cases, where accuracy is not critical and smaller deviations of the 3D reconstruction do not thwart the process of the rehabilitation. Although the accuracy is already quite good under the right conditions, if previously suggested improvements would be implemented the overall performance would be significantly higher.

Acknowledgement: *The authors wish to thank Boris Simončič for his help in the project.*

REFERENCES

1. National Institute of Neurological Disorders and Stroke (NINDS), »Post-Stroke Rehabilitation Fact Sheet,« in NIH Publication No. 08-4846, 2008.
2. American Heart Association, »Heart Disease and Stroke Statistics - 2009 Update,« in American Heart Association, 2009.
3. SUCAR L E, AZCÁRATE G, LEDER R S, REINKENSMEYER D, HERNÁNDEZ J, SANCHEZ I, SAUCEDO P 2008 »Gesture Therapy: A Vision-Based System for Arm Rehabilitation after Stroke«. Proceedings of the First International Conference on Health Informatics, p 107–111
4. OS Platform Statistics [Online]. Available: http://www.w3schools.com/browsers/browsers_os.asp
5. CXCORE Reference Manual [Online]. Available: http://opencv.willowgarage.com/wiki/CxCore
6. CV Reference Manual [Online]. Available: http://opencv.willowgarage.com/wiki/CvReference
7. Experimental and Obsolete Functionality Reference [Online]. Available: http://opencv.willowgarage.com/wiki/CvAux
8. HighGUI Reference Manual [Online]. Available: http://opencv.willowgarage.com/wiki/HighGui
9. KATRAŠNIK J, VEBER M, PEER P 2007 »Using computer vision in a rehabilitation method of a human hand«. Mediterranean Conference on Medical and Biological Engineering and Computing MEDICON, p 947–949
10. Xsens MTi – description and specifications [Online]. Available: http://www.xsens.com/en/general/mti
11. HINESLY D 2008 »Technology in Motion – Assessing the latest motion capture technologies«. Physical Therapy Products, vol. 6.
12. Technology in Motion – description and specifications [Online]. Available: http://www.ndigital.com/lifesciences/certus.php
13. TAO Y, HU H 2006 »3D Arm Motion Tracking for Home-based Rehabilitation«. Proceedings of the 3rd Cambridge Workshop on universal access and assistive technology. Cambridge, U.K., p 105–111
14. TAO Y, HU H 2008 »A hybrid approach to 3D arm motion tracking,« in Transactions of the Institute of Measurement and Control, vol. 30, no. 3/4, p 259–273
15. ETH Zurich, Department of Mechanical and Process Engineering (D-MAVT), ARMin – description and specifications [Online]. Available: http://www.sms.mavt.ethz.ch/research/projects/armin/ therapy
16. Philips Research, Stroke Rehabilitation Exerciser [Online]. Available: http://www.research.philips.com/ technologies/projects/ strokerehab/
17. YAMASHITA A, AGATA H, KANEKO T 2008 »Every Color Chromakey«. Department of Mechanical Engineering, Shizuoka University.
18. SMITH A R, BLINN J F 1996 »Blue Screen Matting,« Proc. SIGGRAPH 2004, p 259–268
19. RYBERG A, CHRISTIANSSON A K, LENNARTSON B, ERIKSSON K 2008 »Camera Modelling and Calibration – with Applications«. Computer vision In-Tech, p 303–333
20. ALTER T D 1992 »3D Pose from 3 Corresponding Points under Weak-Perspective Projection«. MIT, Artificial Intelligence Laboratory, p 1–9
21. FORSYTH D A, PONCE J 2003 Computer Vision: A Modern Approach, Prentice Hall, p 53–94
22. PEER P, SOLINA F 2006 Multiperspective panoramic depth imaging. *In:* Liu J X *(ed)* Computer vision and robotics, Nova Science, p 135–188.
23. Wolfram Research Inc. and Dr. Eric Weisstein Encyclopedia Of Mathematics 2008, p 124, p 1881
24. MONTGOMERY D C, RUNGER G C 2003 Applied Statistics and Probability for Engineers, John Wiley & Sons, p 59–89
25. MOHR R, BOUFAMA B, BRAND P 2006 »Accurate projective reconstruction«. Applications of Invariance in Computer Vision, p 257–276
26. MOHR R, BRAND P 1994 »Accuracy in image measure«. Spie, Videometrics III, p 218–228
27. POGGIO G F, POGGIO T 1984 »The Analysis of Stereopsis«. Annual Review of Neuroscience, vol. 7, p 379–412
28. YU-HUI Z, GUO-QIANG L, YUE-HUI H, WEN-WEN L 2006 »Algorithm and implementation of binocular stereopsis models«. Computer Engineering and Applications, vol. 42, no. 35, p 65–67