Viewing Enhancement in Video-Endoscopy[1]

Dan Koppel and Yuan-fang Wang
Department of Computer Science

Hua Lee
Department of Electrical and Computer Engineering

University of California
Santa Barbara, CA 93106

Abstract. *Video-endoscopy (Figure 1), a mode of minimally invasive surgery, has proven to be significantly less invasive to the patient. However, it creates a much more complex operation environment that requires the surgeon to operate through a video interface. Visual feedback control and image interpretation can be difficult. Poor visual feedback in video-endoscopy prolongs the operation time, increases the risk to the patient, and drives up the cost of health care. It is a major roadblock in replacing the traditional, highly traumatic open surgical procedures with the much less invasive, more patient friendly video-endoscopy, and in training the surgeons to master this new mode of operation. Our research objective is thus to design, code, and validate on real images novel image analysis and rectification algorithms to enhance the visual feedback to the surgeon in video-endoscopy.*
Index terms: computer-assisted medicine, eight-point algorithm, endoscopy, feature tracking

I. Introduction

Our research objective is to develop image analysis and rectification algorithms to enhance the visual feedback to the surgeon in the emerging *minimally invasive surgery* **[2,3]**. There has been a revolution in medical surgery in recent years toward minimally invasive surgery. Minimally invasive surgery reduces the trauma inflicted on the patient during surgery, significantly shortens the time for the patient to recuperate, and lowers the cost of the treatment.

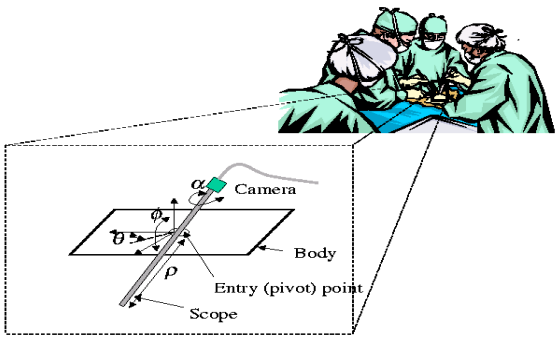

Figure 1: Video-endoscopy and constraints in maneuvering the scope imposed by the pivot point

A key technological advance that has fueled the minimally invasive revolution is video-endoscopy. Endoscopic procedures (Figure 1) are minimally invasive surgical procedures where several small incisions are made on the patient to accommodate surgical instruments such as scalpels, scissors, staple guns, and a video endoscope (also referred to as a scope or a telescope). The scope acquires video images of the bodily cavity that are displayed in real time on a monitor to provide the visual feedback to the surgeon. This setup enables the surgeon to operate instruments through the small incisions, as opposed to a large incision for direct viewing.

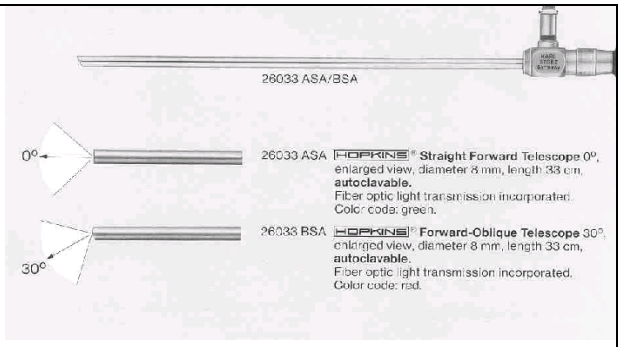

Figure 2: Video scopes of different viewing angles

From our discussions with practicing surgeons and equipment suppliers, we have identified a critical need for image processing assistance to enhance the surgeon's visual feedback in video-endoscopy. The problem is briefly described as follows: When the scope is inserted into a highly constrictive body cavity and is subject to the entry point constraint (Figure 1), blind spots in the view arise. Often times large panning and rotation of the scope is used to eliminate such blind spots. In fact, the scopes are purposely designed with the viewing direction deviating from the length of the instrument to provide an "off-axis" view so that an axial rotation of the scope has a panning effect to enlarge the view volume (Figure 2). However, the view thus acquired is highly non-intuitive, since the physical "up" direction will in general not so appear on the monitor. This effect is called ``*dis-orientation*."

The consequences of dis-orientation are that the surgeon can easily lose the bearing after repeated large movement and rotation of the scope view. This is because in video-endoscopy the body anatomy is not exposed openly. There is no external environment fixture visible in the images to help register the anatomy in the operating room environment. The surgeon has to deduce the scope's bearing based on his/her understanding of the body anatomy.[2] This is difficult if images are not displayed with a strong resemblance to what the surgeon sees in open surgery**.** Hence, solving this problem is part of the general research area of making endoscopic procedures more ``*open-surgery-like*'' and amenable to visual interpretation.

[1] Supported in part by Karl-Storz Imaging, Inc and the University of California Micro Program

[2] An analogy is having a user wear a head-mounted helmet that completely covers the user's field of view. The user sees *only* the computer-generated virtual world. After lengthy immersion, the user loses track of the correlation of the virtual and real worlds.

We propose an image rectification algorithm whose objective is to maintain the surgeon's sense of up and down. In the language of computer graphics [1], the ``head-up'' vector is to be displayed as upward on the screen at all times. The method we have developed and tested extensively on real images can be summarized as follows. We analyze a video stream from an endoscopy procedure to deduce the orientation of the camera and the projection onto the image plane of the physical environment's "up" direction. Panning and rotation of the camera make this projected direction deviate from the screen's upward direction. We then rotate individual video frames by this computed amount, rendering rectified video that properly maintains the head-up direction.

We have broken down the task of computing the rectification angle into three steps. Firstly, 2D features are identified and tracked from frame to frame. Secondly, sets of tracked features are used to deduce the 3D motion undergone by the camera. Finally, the "up" direction, as projected on the image plane, is obtained and the video frames are rotated by this amount.

While the rectification algorithm might appear to be a simple combination of existing techniques, our contribution is in significantly improving the efficiency and robustness of traditional techniques to suit this new application domain. In particular, we reformulated the 2D tracking problem using Fourier analysis and were able to achieve close to 30-fold speed increase over conventional implementation. For 3D tracking, we employ redundancy and robust error norm to significantly improve accuracy and minimize the possibility of loss of track.

II. Mathematical Formulation

The algorithm that processes the video stream comprises essentially three stages. In the first stage, a number of image features are selected and tracked. The features are areas of the image that have a high number of edges or corners with a high intensity contrast. The second stage takes the centers of the tracked 2D features and uses them as inputs to the 8-point algorithm [4] to estimate the camera motion parameters. In the third stage, the direction of the environment's "head-up" vector in the camera's reference frame is inferred. After projecting this vector onto the image plane, the deviation from the y-axis can be computed. Rotation of the image by this amount aligns the "up" direction along the y-axis, achieving rectification.

First stage: 2-D feature tracking

The algorithm starts out by tracking a set of 2-D features as they move from one frame to the next. Here an affine transformation is used to relate the feature's pixel coordinates between frames: $\mathbf{x'} = \mathbf{A}\mathbf{x} + \mathbf{b}$ where $\mathbf{A}$ is a constant matrix and $\mathbf{b}$ is a constant vector. This transformation is equivalent to the following sequence of operations: a) shearing, b) rotation and isotropic rescaling, and c) translation, as can be seen by breaking **A** into two factors:

$$\mathbf{A} = \mathbf{U}diag(\sigma_1, \sigma_2)\mathbf{V}^\mathbf{T} = (\sigma\mathbf{U}\mathbf{V}^\mathbf{T})(\mathbf{V}diag(\varepsilon, 1/\varepsilon)\mathbf{V}^\mathbf{T})$$

where $\sigma = \sqrt{\sigma_1\sigma_2}$ and $\varepsilon = \sqrt{\sigma_1/\sigma_2}$.

The second factor represents step a) and the first factor step b). The reason for decomposing the affine transform into these steps is two-fold. Firstly, finding the six optimal affine parameters can be achieved by searching in each 2D subspace (for steps a-c) separately and sequentially. A tremendous speedup is obtained over a full 6D search. Secondly, for steps b) and c) we have developed efficient search methods. These are now described.

When tracking a feature, the following objective "overlap" function can be defined:

$$f(T) = \sum_{\mathbf{x}\in boundingbox} [I(\mathbf{x}) - J(T\mathbf{x})]^2$$

where I and J are the intensity values of two adjacent frames, T represents the transformation we seek. To simplify the discussion, let us assume that T represents only a translation. In that case, we have:

$$f(\mathbf{b}) = \sum_{\mathbf{x}\in boundingbox} [I(\mathbf{x}) - J(\mathbf{x}+\mathbf{b})]^2$$

If the feature is to be allowed to move by half its width, then **b**'s domain is the same bounding box, but centered at 0. Since all possible values of the argument of J cover an area twice the size of the bounding box, it becomes convenient to extend the sum over this larger area while introducing a factor that "masks" the summand outside the original bounding box. Putting this together, we get:

$$f(\mathbf{b}) = \sum_{\mathbf{x}\in 2*boundingbox} [I(\mathbf{x}) - J(\mathbf{x}+\mathbf{b})]^2 \theta(\mathbf{x})$$

where theta is the mask. Multiplying out, we obtain a standard convolution expression:

$$f(\mathbf{b}) = \sum_{\mathbf{x}\in sweptbox} [I^2(\mathbf{x})\theta(\mathbf{x}) - 2I(\mathbf{x})\theta(\mathbf{x})J(\mathbf{x}+\mathbf{b}) + J^2(\mathbf{x}+\mathbf{b})\theta(\mathbf{x})]$$

The convolution is actually circular, but it is readily seen that the values of **b** which produce a wrap-around effect also give rise to a large f (since a wrapped-around feature in J will be a bad match) and therefore have no influence on the minimization process. Additionally, these values of **b** will be outside **b**'s original domain. Now, the first term is a constant and the second and third terms are convolutions that can be evaluated efficiently for all values of **b**. The use of the Fast Fourier Transform allows this computation to be done in $O(n \log n)$ time, instead of $O(n^2)$ time. Once f has been computed for all values of **b**, the minimum is selected as the solution.

The procedure for handling step b) is much the same, if we perform a certain change of variables. We express the position **x** as a complex number z and use the same expression for *f(T)* as above (except that T now represents rotation and isotropic rescaling). T(**x**) then becomes $z_0 z$ for some complex constant z_0. Thus:

$$J(T\mathbf{x}) = J(z_0 z) = J(\exp(\ln(z_0 z))) = J'(\ln(z_0 z))$$
$$= J'(\ln(z_0) + \ln(z)) = J'(w_0 + w)$$

where *J'* is the composition of the *J* and exp functions and $w = \ln z$ is our change of variables. The key observation to make here is that the argument of *J'* is the *sum* of the bound variable and the free variable and is therefore in the form of a convolution, allowing efficient evaluation through the FFT. More formally, we have:

$$f = \sum_{z \in b.b.} [I(z) - J(z_0 z)]^2 = \sum_{w \in b.b.'} [I'(w) - J'(w_0 + w)]^2 e^{2\mathrm{Re}(w)}$$

The last factor of $e^{2\mathrm{Re}(w)}$ is inserted to preserve the uniformity of the grid spacing in the new variable (analogous to correctly converting differential quantities in the continuous case, upon a change of variables). Multiplying out like before, we again obtain a convolution expression, which can be efficiently evalutated.

As for step a) (shearing), the same complex number approach yields an expression that does not seem to be separable (as above) by a simple change of variable. It is:

$$z \cosh|z_0| + \bar{z}\left(\frac{\bar{z}_0}{z_0}\right)^{1/2} \sinh|z_0|$$

(The presence of both z and $\bar{z}$ causes the problem.) Thus, while the translational and the rotational/rescaling portions of the affine transform are amenable to n*log(n) computation, the shearing portion is not. Therefore, in the interest of efficiency, we implement only steps b) and c).

Second stage: 3-D reconstruction

For the second stage of the algorithm, the point correspondences of at least eight tracked points (obtained in the first stage) are used as inputs. Here we obtain the 3-D transformation that the camera undergoes between the two frames. Two separate steps are involved. Firstly, the "epipolar constraint" [**5**] is used to determine the fundamental matrix using the point correspondences. Secondly, the 3-D transformation is extracted from the fundamental matrix by performing a factorization. In both of these steps an SVD factorization is employed, although in unrelated ways.

In the first step, we use the epipolar condition, that states the following. Given a point correspondence from two frames, $\mathbf{v} = (x, y, -f)^T$ (where x and y are image coordinates and f is the focal length) and $\mathbf{v'} = (x', y', -f)^T$, then $\mathbf{v'^T F v} = 0$. Expressing **F** as a vector: $\mathbf{f} \equiv [F_{11}, F_{12}, F_{13}, F_{21}, F_{22}, F_{23}, F_{31}, F_{32}, F_{33}]^T$ and defining $\mathbf{z} \equiv \mathbf{v} \otimes \mathbf{v'}$ (tensor product), we can rewrite the epipolar condition as $\mathbf{z^T f} = 0$. Since this refers to a specific point correspondence, the **z** needs a label. We can then write the epipolar conditions for all correspondences as a matrix equation as follows. Defining: $\mathbf{Z} \equiv [\mathbf{z}_1, \mathbf{z}_2, ... \mathbf{z}_m]$, we get: $\mathbf{Z^T f} = 0$. To solve this (or obtain the best approximation in the presence of noise), we take the SVD of $\mathbf{Z^T}$ and let **f** equal the right-vector associated with the smallest singular value.

Once **f** is obtained, we re-express it as the matrix **F**. Given that the transformation between frames is given by $\mathbf{x'} = \mathbf{Rx} + \mathbf{t}$ (where **x**, **x'** and **t** are 3x1 vectors and **R** is a 3x3 matrix), the epipolar condition defines **F** as **TR**, where **T** is the 3x3 anti-symmetrized matrix version of **t**. Since we are interested in obtaining **R** for rectification purposes, we need to factor **F**. This can be accomplished with the aid of the SVD of **F**: letting $\mathbf{F} = \mathbf{U\Sigma V^T}$ and noting that $\mathbf{I} = \mathbf{PU^T UP^T}$ for $\mathbf{P} \equiv \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ we get that:

$$\mathbf{F} = \mathbf{U\Sigma(PU^T UP^T)V^T} = \mathbf{(U\Sigma PU^T)(UP^T V^T)} \equiv \mathbf{TR}$$

which is the factorization we seek. (For **T** to be anti-symmetric, the largest singular values need to be equal and the third one should be 0. In the event that this is not true due to noise, the expression for **R** will nevertheless be orthogonal and be an optimal approximation.)

Third stage: image rectification

Finally, once **R** is determined, the orientation (in the camera's coordinate system) of a virtual "up" vector is updated. After projecting the vector onto the x-y plane (the screen), the angle of deviation from the y-axis can be determined. Using this information, the image is rotated to compensate for this deviation angle. This puts the vector (and everything else in the image) in the "up" position as seen on the screen.

Figure 3: Knee mockup with camera and scope

As a final point, it should be mentioned that for the ideal case, the expression from above $\mathbf{Z} \equiv [\mathbf{z}_1, \mathbf{z}_2, ... \mathbf{z}_m]$ can be replaced with no side-effects by the expression $\mathbf{Z} \equiv [\alpha_1 \mathbf{z}_1, \alpha_2 \mathbf{z}_2, ... \alpha_m \mathbf{z}_m]$, where the α's are arbitrary numbers. This follows from the fact that each epipolar constraint equation is homogeneous. In effect, a particular α will act as a weight that that point correspondence has on obtaining the fundamental matrix. Therefore, for real-world data, it makes sense to choose a value for an α that reflects the confidence in that particular point correspondence. Several different criteria can be used here, including using the value of the 2-D feature overlap function (eg. as part of a Boltzmann weighting factor). Another way is to compute **f** in two passes. In the first pass, all the α's are 1 and we obtain a tentative $\tilde{\mathbf{f}}$. For the second pass, the following selection is made: $\alpha_i = h((\mathbf{z}_i^T \tilde{\mathbf{f}})^2)$, where h(x) is a monotonically decreasing function. In our application, we selected h(x) to be a downward step-function, such that the p worst point-correspondences are given a weight of 0 and

the rest are given 1. Or we suppress the potential outliers with zero weight. We used m = 18 and p = 2.

III. Experimental Set-up and Results

Figure 3 shows the setup used to conduct testing. A knee mockup (provided by Karl Storz Imaging) was used as the working environment for the endoscope. To validate the algorithm, we proceeded in two stages. Firstly, testing was conducted on a hybrid of real and synthetic video, where a real image was rotated synthetically. The purpose was to generate sequences with known ground truth. Secondly, real and long video sequences with general camera movement were tested. We described the results below.

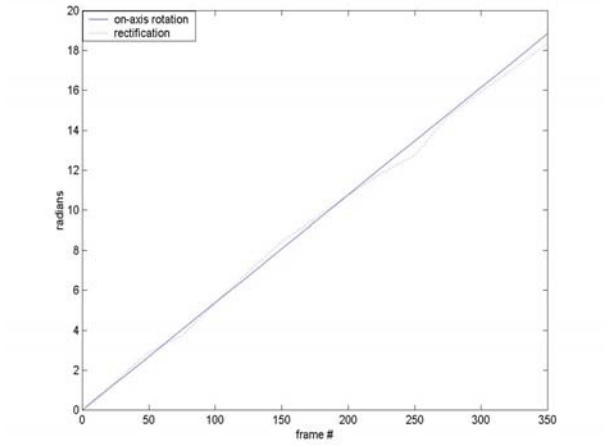

Figure 4: Actual deviation of "up" direction (unrectified) and computed "up": for on-axis rotation

Stage 1: Hybrid of synthetic-real video using a knee mock-up

For this stage, a surgical knee mockup was used (Figure 3). The scope was inserted into a cavity in the knee and a typical image was obtained. This image was then rotated synthetically by known amounts. In typical tests, video executing five rotations was processed by the algorithm. As an initializing step, features of interest are selected by the use of an interest operator. In our case, we selected locations where the determinant of the second moment of the intensity gradient had high values. Additionally, we can adjust two parameters: the number of features used and the number of frames between execution of the 3-D algorithm. For the latter, this need not be one and in fact choosing a larger number is useful for reducing the accumulation of numeric error. Since features may become occluded or exit the field of view, however, this number cannot be too large, if point correspondences are to be maintained. A typical value used here was five. Since this number is small, a low-order predictor can be used to maintain a frame-by-frame generation of rectification angles. Typical number of features used was 16. Figure 4 shows typical results obtained, which show high consistency.

Stage 2: Real video with arbitrary camera motion

In this stage, video obtained from the knee mockup was used. Here the camera was allowed general movement in which all degrees of freedom were varied. Additionally, the camera's position approached the cavity wall significantly, thus simulating real surgical situations. Figure 6 shows an original and rectified sequence. Figure 7 displays the same, but for a different sequence. Figure 5 compares the computed angle of rectification with the amount obtained by (human) analysis of the original sequence (sets of landmarks were carefully examined frame by frame). Given the arbitrary nature of the camera movement (zooming and panning), the correspondence is very good.

Discussion

From our experiments, we conclude that the source of overall error in the algorithm comes from the 2-D tracking. The drift of a bounding box relative to its true position surrounding a feature will impact the computation of the rectification angle. Some robustness can be achieved by over-determining the system of equations (as seen in section II), especially if the drifts are random in direction. Our results show that minimizing these drifts will be helpful to improving rectification for very long sequences. In addition, pre-processing of the images is currently being examined. One example of this is reduction of motion blurring to increase tracking accuracy.

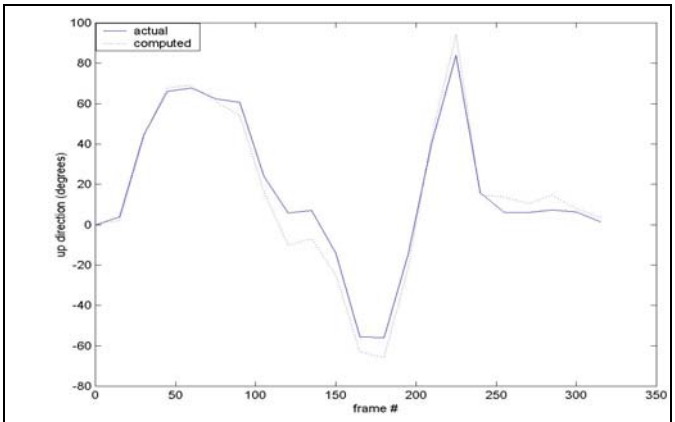

Figure 5: Actual deviation of "up" direction (unrectified) and computed "up": for general camera motion

The timing results for the 2-D tracking using the FFT-convolution method are very positive. Figure 8 compares the new FFT method with traditional software-based methods using hierarchical coarse-to-fine search in the spatial domain. Hierarchical methods construct a pyramid of images of multiple resolutions. The affine parameters are established in coarser resolutions first, and the results are used to guide and refine the search at finer resolutions. We implemented two versions of hierarchical search methods: For regular hierarchical (blue bars in Figure 8), we initially center the search window at the location predicted by the coarser solution. For hierarchical with prediction (red bars in Figure 8), we consider the estimated velocity of the bounding box and center the window where we think the new bounding box will be located. This allows us to employ a smaller initial search window size (since we are most likely already near the true location of the feature), and thus gain in processing speed. Figure 8(a) shows the accuracy results over three typical video sequences. The average drift per frame in tracked feature position over time is shown. Figure 8(b) shows the speed results. As processing speed is not a function of image contents, the comparison is in terms of the feature size. As can be seen from Figure 8, our method achieves better accuracy over traditional methods, *and is significantly faster*. For feature

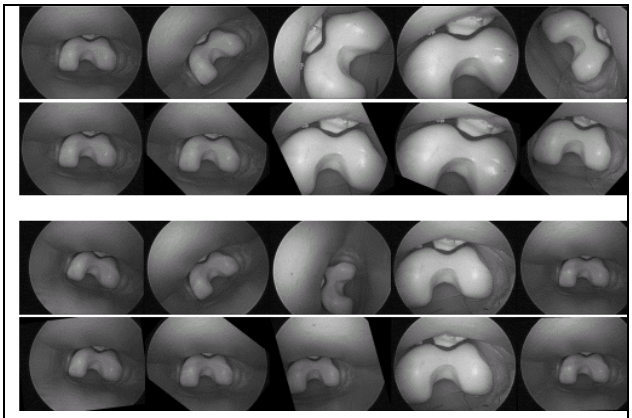

Figure 6: original and rectified sequences (top and bottom rows, respectively) for general camera motion: knee mockup

size of 64 by 64, our method is over one hundred times faster than the hierarchical method. *This speedup is very significant, as the bottleneck in inferring 3D geometry is in 2D tracking* (the 3D reconstruction step described immediately below takes negligible time compared to the 2D tracking step). This improvement is achieved *without* any special hardware (e.g., DSP) acceleration. With hardware acceleration, even greater speedup can be achieved for potential real time or near real time tracking. One last thing to note is that the tracking method is general for all surface types and does not require any special object model to guide the search.

The 3-D component of the algorithm also yields good timing performance. It consumes only 20% of the real-time time requirements.

In the sequences we have analyzed, the deviation of the computed rectification from the known correct amount is approximately 10%. Our current work is focused on reducing this figure.

IV. Concluding Remarks

In this paper, we present our research on enhancing visual feedback to the surgeon in minimally-invasive surgery. A method was presented for estimating the amount of rotation necessary to rectify images forming a video stream to help alleviate the dis-orientation problem in endoscopy. An efficient method of computing 2-D feature tracking was presented. By using a set of epipolar constraints, we obtain the fundamental matrix, which is then factored into a particular form. One of the factors is the rotation matrix that determines the camera's orientation. From this, we update the "up" direction and perform the rectification. By expressing the 2-D tracking objective function as a convolution, we were able to obtain a large speedup over straightforward evaluation. This formulation allows for DSP hardware implementation as well as parallelization (in evaluating 2-D FFT's).

Next steps for us include processing the image stream so that moving objects, such as scalpels, which are not attached rigidly to the rest of the scene, can be segmented out. In this way, only features forming part of the rigid whole will be selected for tracking and the basic assumptions of the algorithm are maintained.

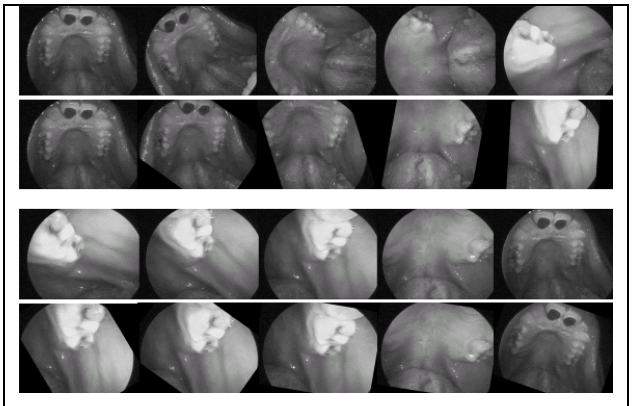

Figure 7: original and rectified sequences (top and bottom rows, respectively) for general camera motion: actual human tissue (mouth)

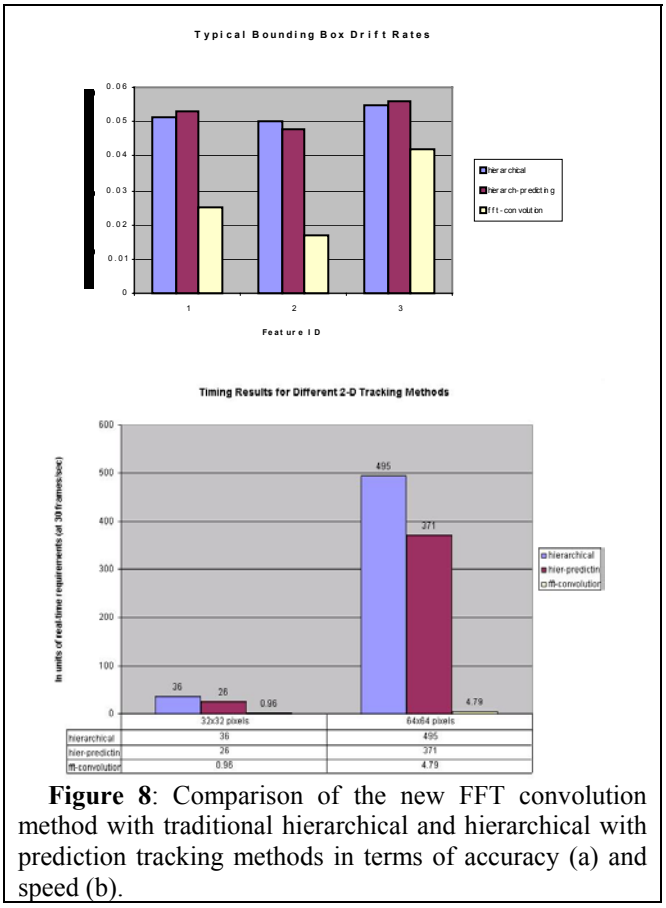

Figure 8: Comparison of the new FFT convolution method with traditional hierarchical and hierarchical with prediction tracking methods in terms of accuracy (a) and speed (b).

References:

1. J.D. Foley, A. van Darn, S.K. Feiner, and J.F. Hughes. Computer Graphics: Principles and Practice, 2nd ed. Addison-Wesley, Reading, MA, 1990
2. J.F. Hulka and H. Reich. Textbook of Laparoscopy, 2nd Ed. W.B. Saunders, Philadelphia, PA, 1994
3. J.G. Hunter and J.M. Sackier (eds.). Minimally Invasive Surgery. McGraw-Hill, New York, 1993
4. H.C. Longuet-Higgins, "A Computer Algorithm for Reconstructing a Scene from Two Projections," Nature, vol. 293, pp. 133-135, Sept 1981.
5. G. Xu and Z. Zhang, "Epipolar Geometry in Stereo, Motion and Object Recognition", Kluwer Academic Publishers, 1996