Department of Cybernetics
Faculty of Electrical Engineering
Czech Technical University in Prague

Jansová Helena

Reconstruction of 3D surface from colonoscopic video

Master thesis

Supervisor: Dr. Ing. Jan Kybic,

Prague 2008

Acknowledgments

I would like to express my gratitude to my supervisor, Dr. Ing. Jan Kybic, for his encouragement and supervision. I would like to thank to Ing. Martin Matoušek Ph.D and Mgr. Daniel Martinec for their kind assistance and sound advice. Special thanks go to my family for their patience and the support they provided me with during my work and throughout my entire life.

Declaration

I honestly declare that I have written the submitted thesis by myself and I have used only the sources (literature, projects, SW etc.) mentioned in the bibliography.

Prague, 23th May 2008

…………………………………….
signature

Abstrakt

Tato práce se zabývá řešením problému automatické 3D rekonstrukce povrchu tlustého střeva z nekalibrované video sekvence pořízené při kolonoskopickém vyšetření. Digitální trojrozměrný model tlustého střeva je v lékařství významným nástrojem při stanovení diagnózy i při lékařské výuce. Vstupní data jsou tvořena nekalibrovanou sekvencí 2D snímků, které jsou zpracovávány metodami založenými na sledování korespondencí z jednotlivých obrázků. Autokalibrační proces je použit k získání vnitřních parametrů kamery. Poté je vyhodnocena trajektorie kamery a pomocí ní pak i výsledný 3D model scény. Algoritmus, který je prezentován v této práci předpokládá, že kamera nevytváří zkosené obrazy, střed projekce leží ve středu obrazu, poměr výšky a šířky pixelů v obrazu je roven 1 a ohnisková vzdálenost má po celou dobu průběhu snímání konstantní hodnotu.

Abstract

This work deals with the problem of automatic 3D reconstruction of colon surface from uncalibrated colonoscopic video sequence. The 3D model of colon is needed in medicine as an essential component of computer-aided diagnosis systems and used as a tool to assist surgeons in visualization, diagnosis and surgical training. The input data consist of uncalibrated 2D image sequence, which is processed using feature correspondence detection and feature tracking techniques. Subsequently, camera autocalibration is performed. Then the camera positions are evaluated and the 3D model of the scene is created. The algorithm provided in this work assumes that the images were taken by a camera with following intrinsic parameters: zero-skew, the principal point is at the image centre, aspect ratio of 1 and fixed focal length.

Table of Contents

1 Introduction

The problem of the 3D reconstruction of the scene has been addressed in many scientific as well as commercial applications for many years and it still remains a challenging ongoing research topic, being termed "the holy grail" in computer vision. At present, obtaining 3D models is required in many applications, such as computer graphics, virtual reality, computer games, movie and entertainment industry, archeology or cultural and historical heritage. In the field of computer vision the 3D reconstruction is used for object recognition or robot navigation. Architectural visualization systems use 3D models in design engineering.

Modeling of three-dimensional scene from a set of digital photographs is called a multiview 3D reconstruction. In case the cameras are fully calibrated (both external and internal parameters of the camera system are known in advance) the reconstruction task becomes simpler. However, the camera calibration involves rather complicated procedure. That is why the recent effort has been to reduce the amount of calibration resulting in solving the problem of the 3D model reconstruction from uncalibrated video sequences. This method allows the user to freely move the camera around an object or scene and take video records. This is one of the most accessible and cheapest ways for 3D model reconstruction.

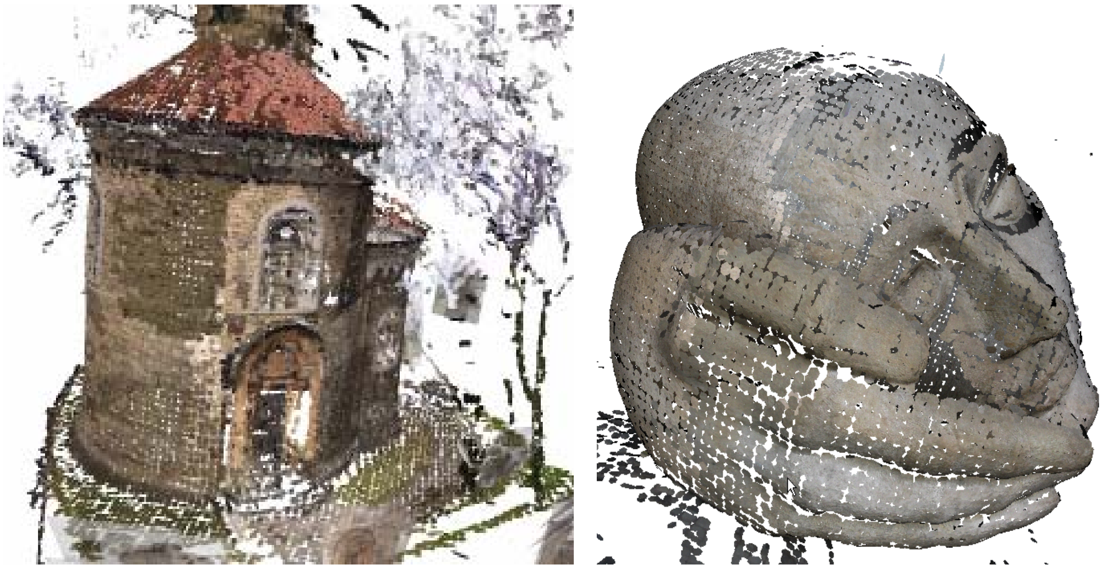


Fig. 1.1. The examples of 3D models reconstructed by research group at the Center for Machine Perception (CMP).

1.1 Problem description

A 3D model of the colon is a needed application in surgical training and diagnosis specification. The aim of this work is to implement a system which performs a sparse 3D reconstruction of the colon surface from a colonoscopic video sequence. The sparse reconstruction is a term for the reconstruction process, the output of which is a 3D model represented by a set of space points, called point clouds.

The input data consist of uncalibrated colonoscopic image sequence divided into particular frames. Uncalibrated image is an image, which was taken by a camera with an unknown position and parameter setting. The set of input images used in this work consists of 30 images with resolution 240x352 pixels representing a consistent part of the original colonoscopic video sequence, available on the enclosed compact disc. Since the distance between camera centers of successive frames in the original sequence is sometimes very small, every other frame from the selected part of the sequence was taken.

Examples of the input frames are shown in Fig. 1.2. In general, quality of the reconstruction depends on the amount and properties of available input data, the scale of the recorded scene, characteristics of the reconstructed structure.

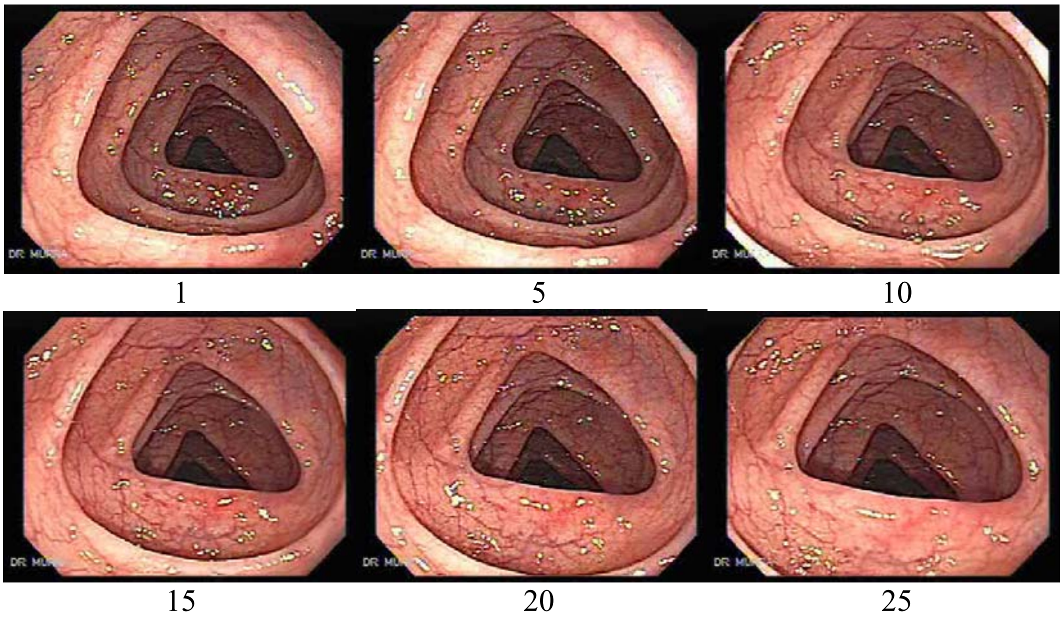


Fig.1.2. The examples of input frames from colonoscopic sequence. Numbers below individual frames denote the order of views within the sequence.

The main steps of the reconstruction process needed to be done are basically as follows. First step is to find correspondences between frames. Using these correspondences, the image pairs can be related by epipolar geometry. The next step includes estimation of the focal length parameter, in order we could subsequently calculate metric camera projection matrices corresponding to all images. Having the correspondences and the camera projection matrices, a sparse structure can be evaluated.

An expected output of this reconstruction process is a sparse colored 3D model of a short part of colon. A criterion of the reconstruction quality will be attained accuracy.

1.2 State of the art

Many different techniques have been developed up to this moment to achieve the 3D reconstruction from image sequences. In this section, I will briefly present a review of several state of the art algorithms coping with the multiview reconstruction from uncalibrated images.

Some reconstruction algorithms work with the constrained nature of the camera motion (e.g. planar camera motion or motion of stereo rig), which facilitates the reconstruction process, as one can utilize some additional constraints on the camera parameters. The example of the reconstruction system, which performs automatic 3D model reconstruction from turn-table sequences, is given in [10]. The input of the system is the uncalibrated image sequence of an object rotating about a single axis. The output is the set of cameras and a 3D texture mapped model of the object. The camera internal parameters are supposed to be fixed. The approach presented in this paper works as follows.

Point tracking is achieved by first obtaining Harris corners for consecutive image pairs and simultaneously computing epipolar geometry and matches consistent with this estimated geometry using a robust estimation algorithm. From these matches, a trifocal tensor is robustly fitted and new correspondences are found by guided matching [52]. For each pair of views the planar-motion fundamental matrix is calculated and subsequently fitted by minimizing the distance of points to epipolar lines. The two parameter reconstruction ambiguity is resolved by specifying camera aspect ratio and parallel scene lines. At this point, the two and three view geometry provides an initial estimate for the camera matrices. An optimal estimate is obtained by nonlinear minimization of the distances between the reprojected 3D points and the 2D corners [53]. Finally, surface generation is performed by a standard marching cubes algorithm [54].

The main benefit of the system described in this paper is that it differs from the previous approaches in the fact that it requires no prior information about the cameras, scene or turntable angles. The system utilizes the constrained type of camera motion providing the additional constraints on the fundamental matrix, trifocal tensor and camera matrices, which are used to determine camera matrices and remove reconstruction ambiguity. Some applications however demand the flexibility of camera motion, for instance the problem specified in this work. Following papers deal with the problem of 3D reconstruction using no special constraints on the camera motion.

In the paper [3], a system which retrieves a 3D surface model from a sequence of uncalibrated images is presented. The suggested algorithm works basically as follows. The Harris corner detector [13] is used to select feature points on images. Matches are determined through normalized cross-correlation of the intensity values of the local neighborhood. Correspondences are determined by RANSAC incorporating 7-point algorithm [1]. Structure and motion is determined in the following steps. Two initial frames are related by epipolar geometry and an initial projective structure is computed. 2D-3D matches corresponding to reconstructed points and image matches in an additional view are inferred and used to compute the projection matrix for the additional view. For every additional view the pose towards the pre-existing reconstruction is determined, and then the reconstruction is updated. Once the cameras have been fully determined the matches can be reconstructed through triangulation.

The next step involves upgrading a projective reconstruction to a metric one by performing auto-calibration process [24]. A search for corresponding points for most of the pixels in the images is performed using a stereo algorithm and the results are refined and completed by combining the correspondences from multiple images. Finally, a textured dense metric 3D surface model is obtained by approximating the depth map with a triangular wireframe [63].

The method described in this paper is capable of reconstructing 3D models by solving the problems of projective reconstruction, auto-calibration and dense correspondence matching. The advantage of the method is that it is able to build metric 3D models without any prior knowledge about the scene or the camera. Moreover, it is not necessary for the initial points to stay visible throughout the entire sequence. The limitation of the system is that it requires the distance between camera centers (the baseline) for successive views to be small. Hence it is not suitable for widely separated views.

Paper [43] describes a method for dense reconstruction of a scene from a set of high-resolution unordered uncalibrated images. In brief the implemented method works as follows. Sparse correspondences are searched for across all pairs of views by matching maximally stable extremal regions (MSERs) [48]. The epipolar geometry for each image pair is estimated using LO-RANSAC [49]. Estimation of a consistent system of cameras is found by factorization method based on [50] refined to be able to cope with missing entries and outliers. Camera auto-calibration is performed using the image of the absolute dual quadric [1]. The auto-calibration is followed by bundle adjustment and radial distortion correction to improve the quality of the sparse model. In order to make dense matching procedure more efficient, image pair rectification is applied and dense matching is subsequently performed as a disparity search along epipolar lines using Confidently Stable Matching (CSM) [50], forming disparity maps per image pairs. The dense point cloud is obtained as the union of the points from all disparity maps. Distribution of points is represented by fish-scales [51]. Various fish-scale sizes are used considering reconstruction details. Finally, texture is mapped on the fish-scales.

The authors of the paper obtained highly accurate 3D models. They presented a new method for the reconstruction of the projective structure and the cameras utilizing a robust and global optimization procedure which avoids propagating reconstruction errors. Moreover, it is not necessary to know an image order. The method does not strongly need the accurate cameras after the camera reconstruction as long as the epipolar geometry is accurate enough for the dense matching. However, a sufficient texture of the reconstructed scene is crucial in order to obtain the geometric accuracy of the final model.

In paper [51], a multiview reconstruction from given pair-wise Euclidean reconstructions up to rotations, translations and scales is introduced. This paper provides a solution to a problem, when no point visible in three or more views is required. The algorithm assumes the input images captured using a camera with focal lengths known up to an unknown overall scale factor. Pairwise image matching is performed with Local Affine Frames [55] constructed on MSER regions, LaplaceAffine and HessianAffine interest points [15]. The outputs of the algorithm are recovered cameras and 3D points.

The suggested algorithm includes following steps. The robust 6-point algorithm [29] is applied to all image pairs in order to estimate corresponding focal-lengths. When the two focal lengths differ, one of the images is rescaled so that the focal lengths become the same for both images. The overall scale of the focal length is then estimated as the mean of the solutions given by the 6-point algorithm weighted by the square of the epipolar geometry support. An epipolar geometry unaffected by a dominant plane is found by method presented in [56] applying the 6-point algorithm as well as the 5-point algorithm [39].

At this stage, two-view partial reconstructions are known, however each reconstruction is generally in a different coordinate system. A system for linear estimation of all cameras together with homographies relating the coordinate systems is constructed. From this system, camera rotations consistent with all reconstructions are estimated in the least squares [57]. Then, all the pair-wise reconstructions are modified according to the new rotations by triangulation using the Sampson's approximation [1]. Bundle adjustment is applied in order to minimize reprojection errors which increased after making rotations consistent. Finally, the refined rotations are used to estimate camera translations and 3D points using Second Order Cone Programming by minimizing the L_∞ – norm [44].

The contribution of this paper is that it introduces a method suitable for a difficult wide base-line set of images. Moreover, the algorithm is able to cope with a situation, when no point is visible in three or more views. This method differs from the previous papers by presenting a new algorithm for evaluating the overall 3D geometry utilizing relative orientations of the cameras (the partial reconstructions are glued via cameras).

The only paper I have found dealing with the problem of 3D reconstruction from colonoscopic video is reported in [8]. This paper presents an algorithm for modeling localized anatomic structures, such as polyps, within a colon. This is done by the following procedure. Camera calibration and distortion correction is performed in order to estimate the intrinsic camera parameters and to correct image distortion. Feature selection, tracking, and matching across multiple images are done using two alternative procedures: continuous tracking and discrete matching. Continuous tracking is performed using the Harris corner detector and FFT-based tracker [58]. Discrete matching is done by SIFT-detector [59].The epipolar geometry for all image pairs is inferred both by a 5-point polynomial algorithm [39] and an 8-point linear algorithm [1] using RANSAC. Then a Levenberg-Marquart (LM) nonlinear iterative optimization procedure [1] is applied to improve the solution.

3D surface for every partial reconstruction is inferred using two different approaches. In the first approach, called sparse landmark approach, the depths of the tracked image feature points are computed to form a sparse depth map and depths of the intermediate pixels are estimated through bi-linear interpolation [60]. In the second approach pixel disparity is computed and 3D depth for every image pixel is estimated. The final step involves a multi-view registration procedure, when partial 2-view models are registered into a complete 3D model through iterative refinement of rigid-body registration parameters [61].

The authors observed that results achieved by using the sparse landmark approach are reasonably accurate for smooth anatomic structures. A more accurate 3D model can is obtained by inferring 3D depth for every pixel in the images. This system presents a reconstruction method suitable for a low number of colonoscopic images, capturing localized structures in the colon. The reconstruction of a long-range colon model is the future plan of the authors.

1.3 Alternative methods for 3D modeling

Modeling of three-dimensional scene is being solved by using various approaches. There are some alternative methods to reconstruction from photographs.

A frequently used approach is represented by modeling methods, which use 3D scanners. These devices collect data on a shape of a real-world object to construct its digital 3D model [7]. Types of 3D scanners can be divided into two main categories, contact and non-contact scanners. While contact scanners explore the subject through physical touch, non-contact scanners utilize some kind of radiation or light and detect its reflection in order to probe an object.

Representatives of 3D non-contact active scanners are for instance a triangulation 3D laser scanner or a time-of-flight 3D laser scanner. The triangulation scanner uses an emitter to radiate a laser pulse on the object and exploits a camera to look for a laser dot on the object surface. This location of the laser dot is subsequently determined using the knowledge of the distance and the angle between the camera and the laser emitter. Unlike this method, the time-of-flight 3D scanner determines the distance of a surface using the information about the amount of time before the emitted light reflected from the object surface is seen by a detector.

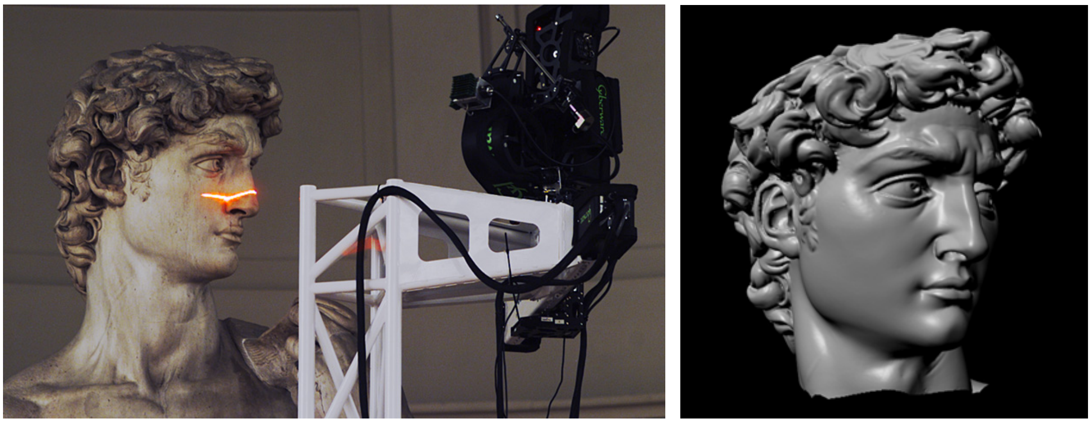


Fig. 1.3. The Digital Michelangelo Project. Left image: scanning the David statue. Right image: a 3D model of David's head.

The ouput of 3D scanners is usually a point cloud (sometimes also providing color information) representing geometric samples of the surface of the analyzed real-world object. The polygonal 3D model can be then obtained by using various reconstruction algorithms. An interesting research project, which has been performed, is for instance The Digital Michelangelo Project [46], which aimed to create 3D models of Michelangelo's statues in Florence using the laser triangulation scanner. Another notable project is Plastico di Roma antica, dealing with digitizing a 3D model of Rome [47].

The 3D scanners can provide very accurate 3D models and they are extensively used in the industrial design, the entertainment industry or cultural heritage. However there is difficulty in modeling shiny or transparent objects. Another disadvantage of these systems is that they are built around specialized hardware resulting in high costs.

In medicine, the problem of the 3D modeling is often solved by using systems such as Computed Tomography (CT) or Magnetic Resonance Imaging **(**MRI). Both CT and MRI scanners can generate multiple two-dimensional cross-slices, which are stacked on the top of each other to create a 3D volume. Extraction of a surface from the volume data is usually performed by some kind of computer graphics algorithm. The Marching Cubes algorithm [64] is a common technique.

The inverse problem to 3D reconstruction from colonoscopic video is solved by a method called virtual colonoscopy. Virtual colonoscopy is a medical imaging procedure that employs CT or MRI volume data and visualization techniques to generate a fly-through inside a colon. It is basically an alternative diagnose technique to the conventional colonoscopy. The advantages of virtual colonoscopy are that it is a noninvasive and painless procedure, thus more comfortable than conventional colonoscopy, with a small risk of perforation. On the other hand, disadvantages are the cost, lower level of details than a conventional colonoscopy shows and missing information about the true color of the structure. The example of a volume rendering system generating images for virtual colonoscopy is presented in [9]. The illustration of human colon visualized by means of virtual colonoscopy technique can be seen in Fig. 1.4 [6].

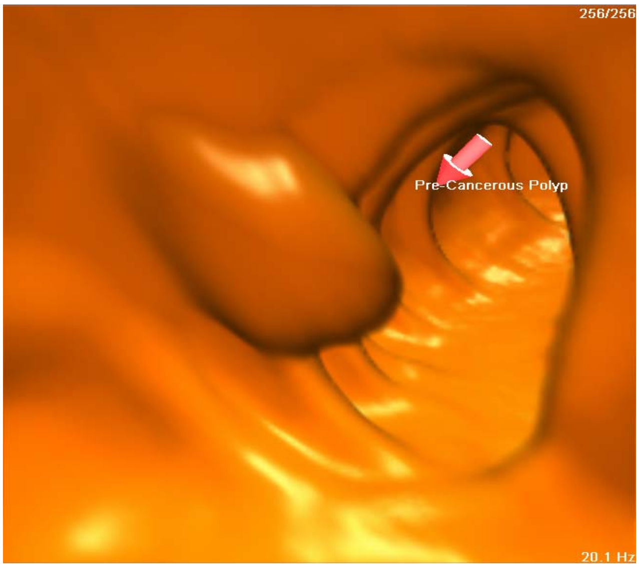


Fig.1.4. The example of a human colon visualized with system VolVis [6].

1.4 Overview of the suggested method

The method suggested in this work uses some techniques from the above described state of the art algorithms. The reconstruction method works basically as follows.

At first, the Harris corner detector is used to identify feature points on images. Feature tracking is achieved by using two similar methods – correspondence-based technique and template matching technique. Both methods determine matches through normalized cross-correlation function of the intensity values of local neighborhoods. Afterwards, the robust normalized 8-point algorithm is used to estimate epipolar geometry and matches consistent with this estimated geometry. Having image correspondences, the robust 6-point algorithm is applied to all image pairs in order to estimate a focal-length of colonoscopic camera corresponding to a maximum of a focal-length distribution curve. Afterwards all camera internal parameters are known, the calibrated five-point polynomial algorithm is applied to all image pairs to infer relative orientation between view pairs. A consistent scale of pair-wise metric reconstructions is estimated stepwise over three views and global cameras are calculated. Finally, a colored sparse structure is generated.

This work is organized as follows:
In *Chapter 2* implemented feature detection and tracking methods are described. In *Chapter 3* basic concepts of camera geometry, epipolar geometry and structure computation are explained. A robust 8-point algorithm to estimate the fundamental matrix is presented in *Chapter 4.* Auto-calibration problem is solved in *Chapter 5.* In *Chapter 6* the solution for a multiview reconstruction based on pair-wise metric reconstructions obtained via calibrated five-point polynomial algorithm is presented. In *Chapter 7* the suggested algorithm is tested on synthetic data. *Chapter 8* presents reconstruction results using real colonoscopic data. *Chapter 9* concludes the work.

2 Feature detection and tracking

Feature detection and tracking is a crucial task, as the whole 3D reconstruction algorithm relies on determining a location of a set of points in the image and on the precise tracking of those moving points through the image sequence. The inaccuracy of feature trajectories significantly affects the final 3D model.

A concept of feature in computer vision refers to an "interesting" part of the images. One of the most important requirements for a feature is that it can be differentiated from its neighborhood. Classically, features are selected according to a measure of texturedness and cornerness, which is based on derivative values in more that one spatial direction. Types of features can be classified as corners (interest points), edges, regions of interest and ridges.

A very large number of feature detectors have been developed, varying generally in the types of features detected and in the computational complexity. From the other point of view detectors can be divided into groups according to their invariance to rotation, scale, illumination variation, blur, affine tranformation etc. An important class is formed by affine-invariant detectors including e.g. Harris-affine region detector [65], Hessian-affine region detector [15] or Maximum Stable Extremal Regions (MSER) detector [48]. In this work, I use feature point detector, called Harris corner detector [13].

Feature tracking is a basic operation in computer vision. I implemented two basic feature tracking techniques - correspondence based technique and template matching technique. The advantage of the first technique is that the same feature can be detected relatively reliably and consistently across many frames. The disadvantage is that correspondence errors can be large. The second technique enables to avoid large errors, but at the expense of the accuracy of tracking because the features tend to drift [16].

2.1 Detection of feature points

As a feature detector, I use Harris corner detector [13]. The Harris corner detector is a popular feature point detector as it is reasonably invariant to rotation and different sampling and quantization, illumination variation, small changes of scale and small affine transformations [14]. Moreover, it is easy to implement. The disadvantage is that it is not invariant to more distinctive scale changes and affine transformation. Harris corner detector is based on the local autocorrelation function - the features are evaluated by considering the autocorrelation function of small region around an image point. Theoretical background of the Harris corner detector derivation is basically as follows [14] [13].

Given an image function $I(x, y)$ at point (x, y) and a shift $(\Delta x, \Delta y)$, then the auto-correlation function [14] is given as

$$c\,(x,y;\Delta x,\Delta y) = \sum_{\mathbf{W}} g(x_i,y_i)\big(\,I(x_i,y_i) - I(\,x_i+\Delta x, y_i+\Delta y)\,\big)^2, \qquad (2.1)$$

where $\mathbf{W}(x, y)$ is a window centered at point (x, y), $(x_{i,} y_i)$ are the points in the window $\mathbf{W}$ and $g(x_{i,} y_i)$ is Gaussian weighting factor $\exp((x_\mathrm{i} - x)^2 - (y_\mathrm{i} - y)^2)\,/\,2\sigma^2$. The shifted image function is approximated by the first-order Taylor expansion:

$$I(x_i + \Delta x, y_i + \Delta y) \approx I(x_i, y_i) + I_x(x_i, y_i)\,\Delta x + I_y(x_i, y_i)\,\Delta y \tag{2.2}$$

$$= I(x_i, y_i) + \left[I_x(x_i, y_i), I_y(x_i, \mathrm{y}_\mathrm{i})\right]\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}\ , \tag{2.3}$$

where I_x, I_y are partial derivatives in x and y, respectively. Substituting equation (2.3) into equation (2.1), we approximate the autocorrelation function by a quadratic function:

$$c(x, y; \Delta x, \Delta y) \approx \sum_{\mathbf{W}} g(x_i, y_i)\left(\left[I_x(x_i, y_i), I_y(x_i, y_i)\right]\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}\right)^2 \tag{2.4}$$

$$= [\Delta x \Delta y]\sum_{\mathbf{W}}\left(g(x_i, y_i)\begin{bmatrix} I_x(x_i, y_\mathrm{i})^2 & I_x(x_i, y_i) I_y(x_i, y_i)\\ I_x(x_i, y_\mathrm{i}) I_y(x_i, y_i) & I_y(x_i, y_i)^2\end{bmatrix}\right)\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix} \tag{2.5}$$

$$= [\Delta x \Delta y]\begin{bmatrix} A(x, y) B(x, y)\\ B(x, y) C(x, y)\end{bmatrix}\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix} \tag{2.6}$$

$$= [\Delta x\, \Delta y]\,\mathbf{Q}(x, y)\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}\ , \tag{2.7}$$

where $\mathbf{Q}(x,y)$ is a symmetric positive semi-definite matrix and its eigenvalues λ_1, λ_2 capture the intensity structure of the windowed image region.

There are three cases to be considered. If both eigenvalues are small, there is a little change in $c(x,y)$ in any direction and thus the region is flat, having approximately constant intensity. If one eigenvalue is small and the other one large, then $c(x, y)$ significantly changes only in one direction indicating an edge region. If both eigenvalues are large, then $c(x,y)$ changes considerably in all directions indicating a corner region. Examples of these three instances are given in Fig. 2.1 [14]. Autocorrelation function is represented by ellipses. Elongation and size of the ellipse are given by eigenvalues of $\mathbf{Q}(x,y)$.

As was shown above, a corner response signal is determined by considering the eigenvalues of the matrix $\mathbf{Q}(x,y)$. Harris [13] suggested a corner function, also termed cornerness:

$$H(x, y) = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 \tag{2.8}$$

with the constant $k = 0.04$. This constraint can also be expressed as[1]

$$H(x, y) = det(\mathbf{Q}(x, y)) - k\, trace^2(\mathbf{Q}(x, y)) \tag{2.9}$$

[1] *det*(**Q**) is a notation for determinant of the matrix **Q**. *trace*(**Q**) is defined to be the sum of the elements on the main diagonal of **Q**

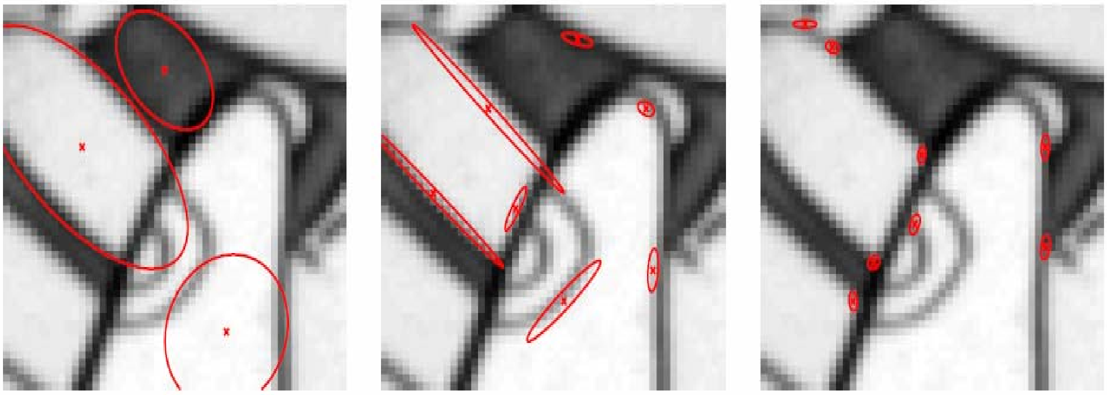


Fig. 2.1: Ellipses with equation $[\Delta x\, \Delta y]\mathbf{Q}(x,y)[\Delta x\, \Delta y]^{\mathrm{T}} = 1$ [14]. Left image: both eigenvalues are small - flat region; Middle image: one eigenvalue is small, the other one large - edge; Right image: both eigenvalues are large – corner.

Evaluating the cornerness function over the whole image, we get **H.** Corner points are defined as local maxima of **H**, see Fig. 2.2 [14]. Selection of corner points can be controlled by defining a minimal threshold corner value t_H, which determines whether a local maximum is still evaluated as a corner, or by setting a various value r_H for radius of region considered to find local maxima (e.g. if r_H =1 px[2] then size of the local region is equal to 2 r_H +1 px). Sub-pixel precision of corner features can be achieved by finding local maxima on quadratic surface interpolating the cornerness function.

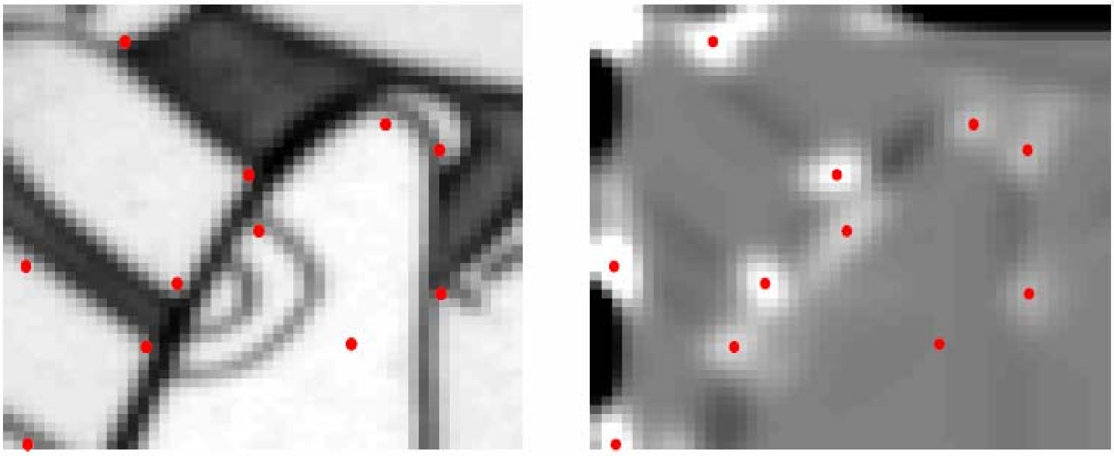


Fig. 2.2: Image **I**(x,y) (on the left) and its corner function **H**(x,y) (on the right) [14].

[2] px is used as an acronym for pixel (picture element).

2.2. Feature tracking

Feature tracking is a procedure where a position of particular feature is identified across two or more images. The set of image coordinates representing the position of the feature across the set of views is referred to as a track. Feature tracking is the method for extracting motion information from a set of images.

Since the input images are a part of a video sequence, it is possible to assume the consecutive images do not differ too much. Therefore we suppose that the local neighborhood of image points representing the same scene point looks similar for the close frames. This assumption allows us to use the intensity normalized cross-correlation function as a measure for matching the corresponding features, since it is invariant to image translation and offsets in the intensity values and can therefore rather efficiently cope with small variations in a camera pose and illumination. I applied two different tracking approaches, described bellow.

2.2.1 Correspondence-based technique

In the first approach, so called correspondence-based technique [16], feature points are detected in each image of the sequence independently using the Harris detector [13]. Each pair of successive images is then processed in order to establish putative matches between corresponding sets of features. Matching is done by normalized cross-correlation (NCC).

Normalized cross-correlation determines the strength and the direction of linear relation between the brightness values. Let b_{1i} and b_{2i} be the brightness values in matrices representing the feature point neighborhoods with n pixels and let $\bar{b}_1$ and $\bar{b}_2$ represent the mean values over the appropriate surroundings. Then the following formula denotes the value of the normalized cross-correlation function r_{ncc}, also called a linear correlation coefficient [11].

$$r_{ncc} = \frac{\sum_{i=1}^{n}\left(b_{1i}-\bar{b}_1\right)\left(b_{2i}-\bar{b}_2\right)}{\sqrt{\sum_{i=1}^{n}\left(b_{1i}-\bar{b}_1\right)^2}\sqrt{\sum_{i=1}^{n}\left(b_{2i}-\bar{b}_2\right)^2}} \tag{2.10}$$

In case that the values are exactly related by linear function with positive/negative derivative, $r_{ncc}(b_1,b_2) = +1/-1$. Otherwise $-1 < r_{ncc}(b_1,b_2) < +1$.

The input data represented in RGB notation are converted to grayscale intensity image and the correlation coefficient is calculated for grayscale images. The conversion is done by eliminating hue and saturation information while retaining the luminance (by Matlab function *rgb2gray*).

Each feature point is surrounded by a small square window. The NCC is computed using every possible match combination. That means, for every feature point $n_1 = 1 \ldots N_1$ in the first image the NCC values with every feature point $n_2 = 1 \ldots N_2$ in the second image of the pair are calculated. The NCC values are placed in a matrix **C** with a size $N_1 \text{ x } N_2$, its element $\mathbf{C}(n_1,n_2)$ indicates similarity relation between the n_1 neighborhood in the first image and n_2 neighborhood in the second image.

Putative matches are defined as feature pairs (n_1,n_2), which satisfy a symmetry condition [11]

$$\mathbf{C}(n_1,n_2)=\underset{n\in 1\ldots N_1}{max}\mathbf{C}(n_1,n_2)=\underset{n\in 1\ldots N_2}{max}\mathbf{C}(n_1,n_2). \tag{2.11}$$

In addition, a threshold T_{ncc} for the correlation coefficient values should be specified. Only putative matches with

$$\mathbf{C}(n_1,n_2)>T_{ncc} \tag{2.12}$$

are retained. Implementation of the threshold constraint for correlation coefficient helps to avoid correspondence errors.

As the motion between the two images is supposed to be small, the location of the feature points should not change much between two consecutive views. This assumption is therefore used to reduce the combinatorial complexity of the matching and only features with similar coordinates in both images are compared. For example, a feature point with image coordinates (*x,y*) is only compared with the feature points of the other image with image coordinates located in the interval

$$([x-w,x+w],[y-w,y+w]), \tag{2.13}$$

where value w is set with regard to expected size of the movement.

2.2.2 Template matching technique

The second approach, called a template (block) matching [62], extracts a set of Harris corners from the first frame only. The position of these corners in subsequent frames is found by using an exhaustive search method inside a suitably sized window. This window size is defined with regard to expected size of the movement, analogous to correspondence-based technique.

The principle of the template matching method is illustrated in Fig. 2.4. Each feature point is again supposed to be a center of a small window surrounding the feature point. For each detected corner with coordinates (*x,y*), the corresponding match is supposed to be found in the other image inside the area defined by equation (2.13). The exhaustive search is realized within this region (blue square), using pixel-by-pixel shifting of the feature neighborhood both in horizontal and vertical direction (green square). The calculation of the brightness similarity by the normalized cross-correlation between the feature neighborhood in the first image and the actual neighborhood in the other image are evaluated for all the shifts. For this purpose, RGB images are again converted to a grayscale. The maximal value determines the location of the corresponding feature point in the new frame.

If the value of the correlation coefficient is lower than a specific threshold T_{ncc}, as in the previous described method, the feature point is marked as “an uncertain point” and traced no more. This should reduce point trajectory inaccuracy because uncertain points are often noisy.

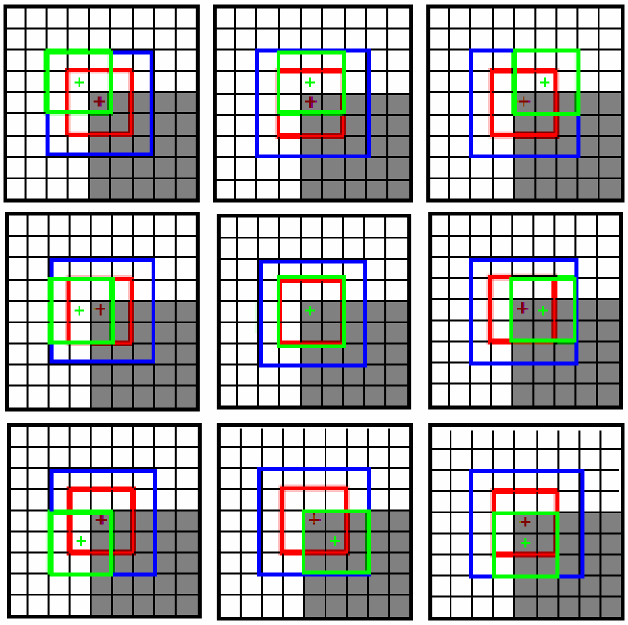


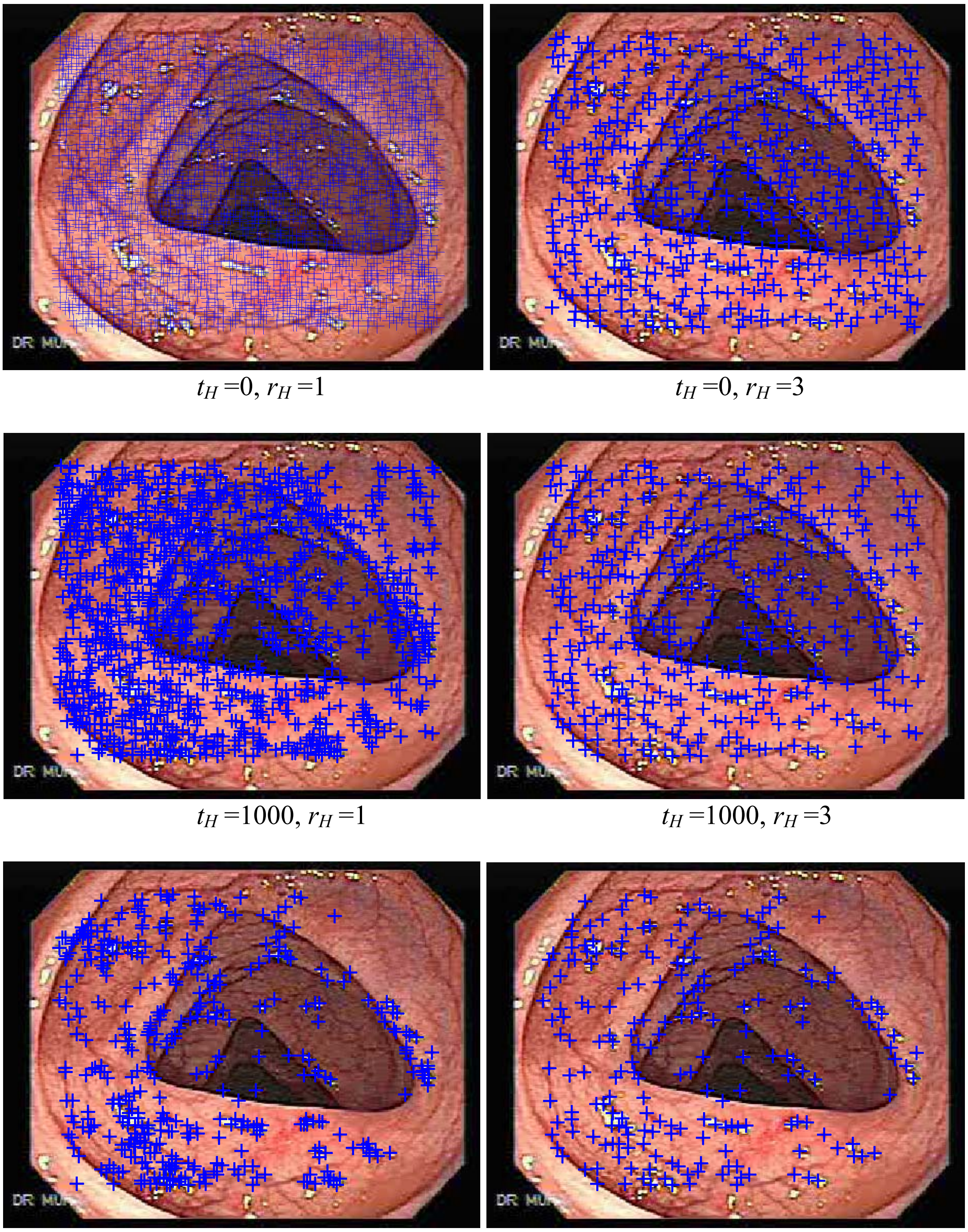
Fig. 2.4: Schema of the template matching technique using exhaustive search method. Red window - the location of a feature point neighborhood in the first image. Blue window - the area inside which the exhaustive search is performed. Green window – shifted neighborhood of the potential correspondence, used to calculate NCC value.

2.3 Experiments on colonoscopic data

Some experiments on colonoscopic images using various parameter settings were performed. Examples of corner points detected in the input colon images are shown in Fig. 2.5. As can be seen, the number of detected corner points depends on the values of parameter t_H and r_H. The largest number of detected feature points is obtained for t_H =0, r_H =1. In the algorithm implemented in this work, this parameter setting is used, since it gives the largest initial set of image points suitable for tracking.

Demonstration of feature tracking on a sample pair of images performed by using correspondence-based technique and template matching technique is shown in Fig. 2.6 and in Fig. 2.7. The parameter setting was set to be the same for both methods to provide a comparison. Basically, the both methods give similar results. The template matching technique generally gives larger number of putative matches. However, the running time of the template matching method is significantly longer, since an exhaustive search requires a relatively long computing time.

Typical parameter values of tracking techniques applied to colonoscopic images in this work are as follows: minimal threshold for the correlation coefficient T_{ncc} = 0.7, maximum search distance for matching w = 10 px (as the expected size of camera movement between successive frames is relatively small), window size for correlation matching w_{ncc}= 9 px.



t_H =0, r_H =1

t_H =0, r_H =3

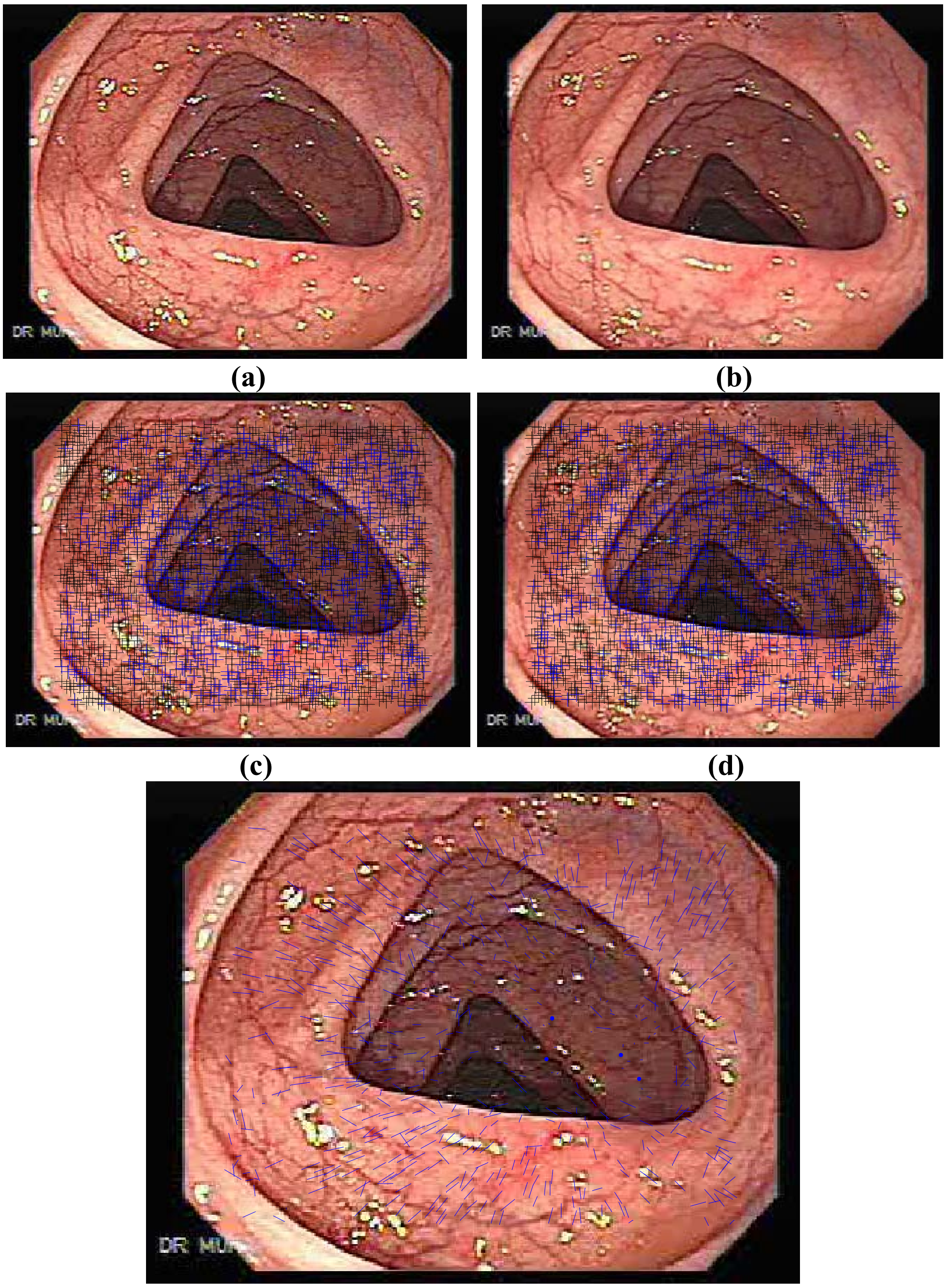
t_H =1000, r_H =1

t_H =1000, r_H =3

t_H =3000, r_H =1

t_H =3000, r_H =3

Fig. 2.5. Detection of Harris corner points on a colonscopic image – individual corners are marked with blue crosses. Results for various levels of minimal corner threshold value t_H and parameter r_H defining a radius of region considered to find local maxima are presented.

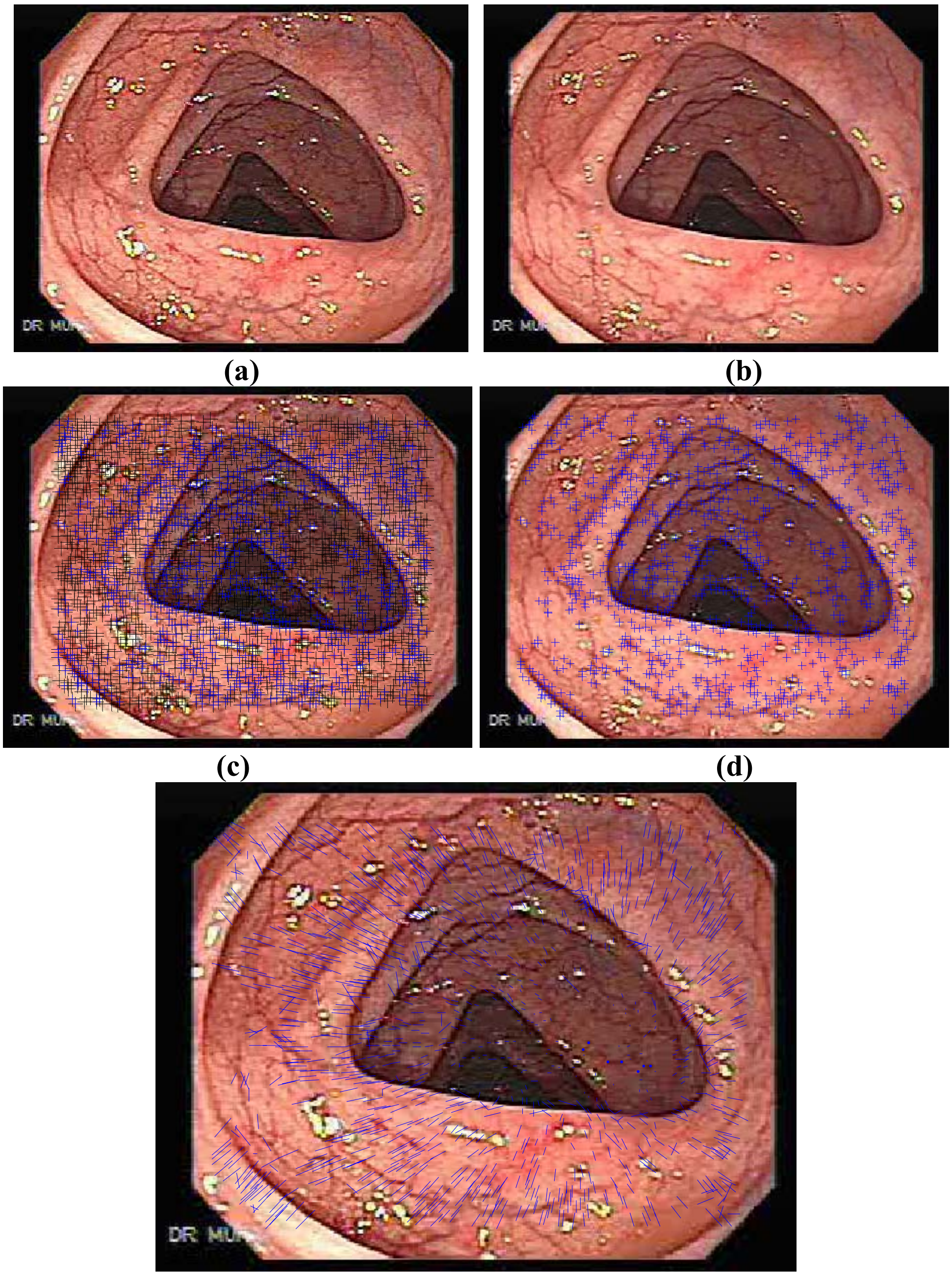


(a) **(b)**

(c) **(d)**

(e)

Fig 2.6. Feature tracking performed on a sample image pair by using correspondence-based technique. (a) (b) Input image pair. (c) (d) Detected Harris corners pictured in the image pair. Blue crosses represent a subset of corners which were identified as putative matches, while the rest of corners are marked with black. (e) Putative matches superimposed on the left image shown by the blue line linking the location of the feature points in the left and right images. Parameter settings: radius of the local region for corner detection r_H = 1 px; corner threshold value t_H = 100**,** minimal threshold for the correlation coefficient T_{ncc} = 0.7*,* maximum search distance for matching w = 15 px. Window size for correlation matching w_{ncc}= 9 px.



(a) **(b)**

(c) **(d)**

(e)

Fig 2.7. Feature tracking performed on a sample mage pair by using template matching technique. (a) (b) Input image pair. (c) Detected Harris corners pictured in the left image. Blue crosses represent a subset of corners which were identified as putative matches, while the rest of corners are marked with black. (d) Identified putative matches in the right image marked with blue. (e) Putative matches superimposed on the left image shown by the blue line linking the location of the feature points in the left and right images. Parameter settings: radius of the local region for corner detection $r_H = 1$ px; corner threshold value $t_H = 0$**;** minimal threshold for the correlation coefficient $T_{ncc} = 0.7$, maximum search distance for matching $w = 15$ px (the radius of the window inside which the exhaustive search is performed). Window size for correlation matching $w_{ncc}= 9$ px.

3 Two-view geometry

The two-view geometry is the geometry of two perspective views. In this chapter, basic concepts of camera geometry, epipolar geometry and structure computation useful for further work are explained. Note that homogenous representation of points and lines is used [1].

To comprehend relations of the two-view geometry, we need to understand camera geometry first [1]. A camera maps the 3D world into a 2D image. In this work, the camera performing a central projection of points in space onto a plane is assumed. Mathematical model of this camera is represented by a projection camera matrix. This matrix holds information about the camera position and orientation in space during acquisition as well as information about internal camera parameters, such as focal length or resolution.

The epipolar geometry is the projective geometry between two views [1]. Assuming that the cameras are sufficiently well approximated by the pinhole camera model, the epipolar geometry represents geometric relations between 3D points and their projections onto the two views in distinct positions.

Having computed camera projection matrices, 3D points of a reconstructed scene can be subsequently calculated by a so called linear triangulation method [1], utilizing the knowledge of the camera projection matrices and image correspondences.

3.1 The projective camera model

The projective camera model is a generalized form of the pinhole camera, including a rigid transformation of the world coordinates and translation and scaling of the image coordinates. Starting from the pinhole camera model, shown in Fig. 3.1 [1], we will describe the projective camera geometry.

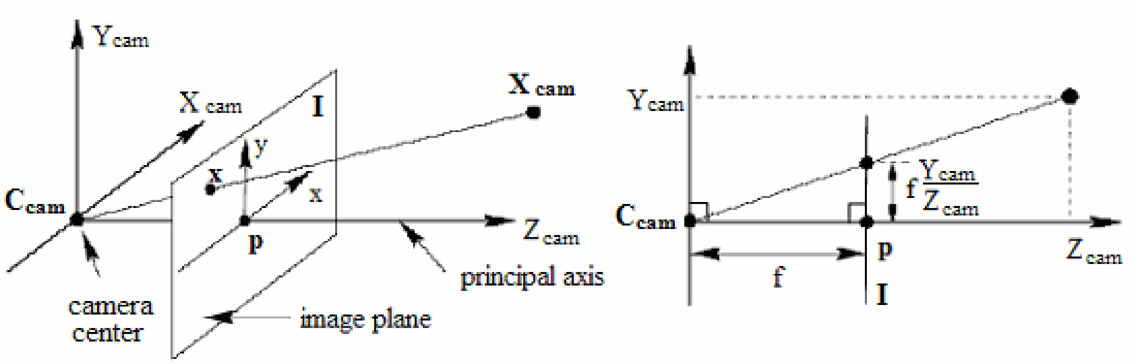


Fig. 3.1. Pinhole camera geometry

Let us suppose the center of projection $\mathbf{C}_{\mathbf{cam}}$ at the origin of a Euclidean coordinate system, the image plane **I** with the principal point **p,** focal length f and the point in space $\mathbf{X}_{\mathrm{cam}}$. The process of projection the space point $\mathbf{X}_{\mathbf{cam}} = (X_{cam}, Y_{cam}\ , Z_{cam})$ into the image point $\mathbf{x} = (x,y)$ is described by the intersection of a line passing through the space point

X$_{\mathbf{cam}}$ and the camera center **C**$_{\mathbf{cam}}$ with the image plane **I**. Using similar triangles we derive that this projection is described by mapping

$$x = f\frac{X_{\text{cam}}}{Z_{\text{cam}}},\ y = f\frac{Y_{\text{cam}}}{Z_{\text{cam}}}\ . \tag{3.1}$$

If the world point and the corresponding image point are represented by homogenous coordinates, the central projection may be written as

$$\begin{pmatrix} fX_{\text{cam}} \\ fY_{\text{cam}} \\ Z_{\text{cam}} \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X_{\text{cam}} \\ Y_{\text{cam}} \\ Z_{\text{cam}} \\ 1 \end{pmatrix}, \tag{3.2}$$

which may be expressed in the form

$$\mathbf{x} = \mathbf{K}\,[\,\mathbf{I} \mid \mathbf{0}]\,\mathbf{X}_{\mathbf{cam}}, \tag{3.3}$$

where **K** is a 3 x 4 homogenous matrix.

So far we have supposed that the space point **X**$_{\mathbf{cam}}$ is represented in the coordinate frame defined by the camera, that is in the camera coordinate frame, in order to illustrate more easily the basic concept of the camera model. In practice, we need to express the points in space in a different coordinate frame, in the world coordinate frame, which is related to the camera coordinate frame by rotation **R** and translation **t**, see Fig. 3.2 [1].

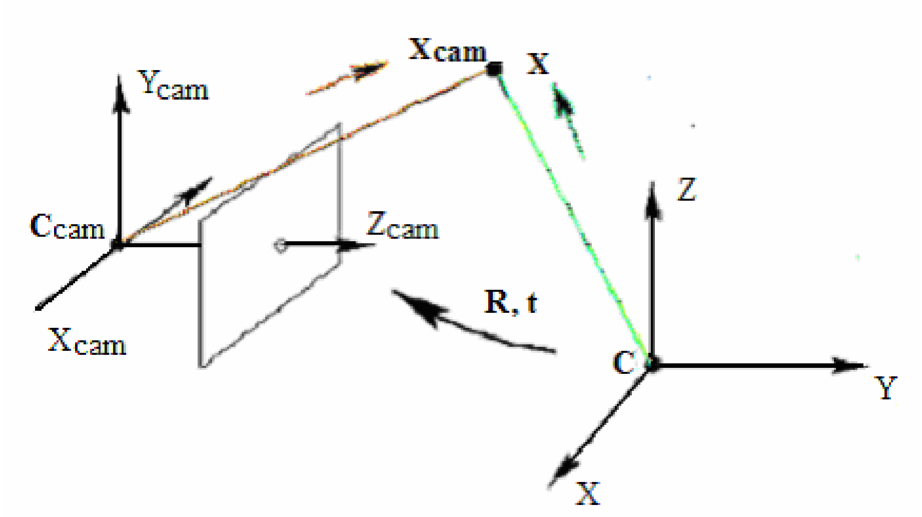


Fig. 3.2. Transformation between the camera and the world coordinate frame

The transformation between coordinate frames can be defined as

$$\mathbf{X_{cam}} = \begin{bmatrix} \mathbf{R} & \mathbf{-RT} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \mathbf{X} \tag{3.4}$$

where $\mathbf{X_{cam}}$ is the space point expressed in the camera coordinate frame and **X** is the same space point represented in the world coordinate frame. **R** is a 3x3 rotation matrix and **T** represents the coordinates of the camera center in the world coordinate frame. Putting this together with (3.3), we get:

$$\mathbf{x = K\ [R|\text{-}RT]\ X = K\ [R|\ t]\ X,} \tag{3.5}$$

which can be written in the concise form as

$$\mathbf{x = PX,} \tag{3.6}$$

where parameter **t** = **-RT** represents a translation vector. The matrix **P** = **K[R|t]** is the projection camera matrix consisting of a 3x3 camera calibration matrix **K**, representing the internal camera parameters, and 3x4 matrix **[R|t]**, which holds information about external camera parameters – exterior orientation. For the projective camera model, the calibration matrix has a general form

$$\mathbf{K} = \begin{bmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}, \tag{3.7}$$

where $\alpha_x = fm_x$ and $\alpha_y = fm_y$ with constants m_x and m_y indicating the number of pixels per unit distance in the x and y directions and parameter f marks the focal length of the camera. Thus α_x and α_y represent the focal length in terms of the scale factor used in the x and y directions. Similarly, $x_0 = m_x p_x$ and $y_0 = m_y p_y$ are the coordinates of the principal point ***p*** in the image coordinate system, as shown in Fig. 3.3. These parameters are equal to zero if the origin of the image coordinate system is at the principal point, as in the case described by (3.2). Nevertheless, in practice it is often not so [1].

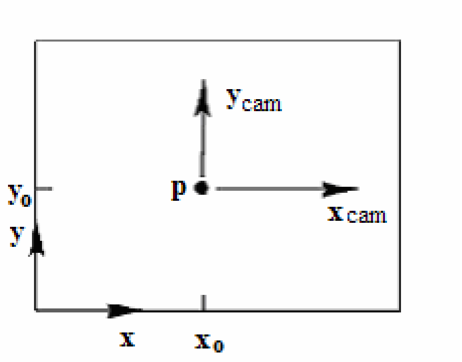


Fig. 3.3. The relation between camera (x_{cam}, y_{cam}) and image (x, y) coordinate systems.

Parameter *s* is known as a skew parameter characterizing the angle between the axes of the image coordinate system. If the axes are vertical, it is equal to zero. The equation (3.6) is an important formula defining the mapping between the world point **X** and the image point **x** by the projective camera.

3.2 Epipolar geometry

As was mentioned above, the epipolar geometry is the intrinsic projective geometry between two views. It is represented by a fundamental matrix and it is dependant only on the relative camera position and internal camera parameters, but not depending on the structure of the observed scene alone (that is on the choice of world coordinate frame). In this section, the basic relations of the epipolar geometry are explained [1].

At first, let us suppose a point **X** in a 3D space, which is imaged in two views, corresponding to point **x** in the first image **I** and point **x′** in the second image **I′**. Let **P** and **P′** denote cameras with corresponding camera centers **C** and **C′**. As shown in Fig. 3.4, the image points, the space point **X** and the camera centers are coplanar, lying in the plane Π.

The ray defined by the camera center **C** and the image point **x** is projected to in the image **I′** defining a line **l′.** This line is known as an epipolar line and the point **x′** must lie on this line **l′**. Similarly, the point **x′** and the camera center **C′** induce an epipolar line l in the image **I**.

A line **b** joining the camera centers is called a camera baseline. The camera baseline intersects image planes at points **e** and **e**′, called epipoles. The epipole in one view is the image of the camera center of the other view. All epipolar lines intersect in the epipole [1].

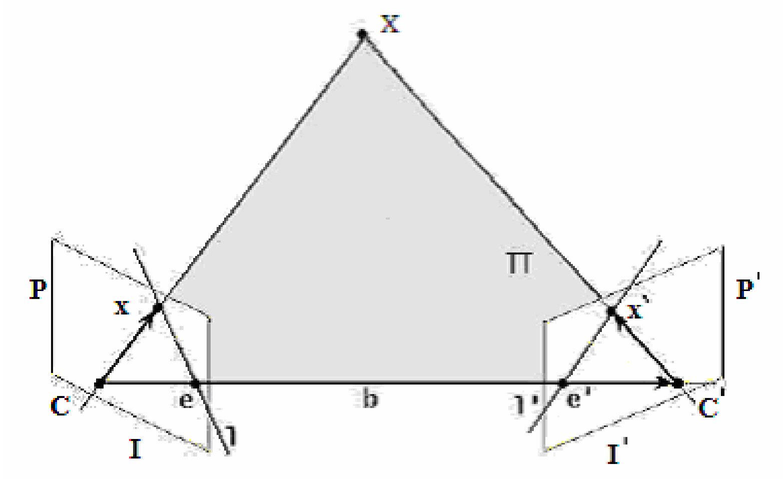


Fig. 3.4: Two-view geometry

The above described transformation between the image point and its epipolar line is linear and can be expressed by

$$l' = \mathbf{F}\,\mathbf{x}\,, \qquad (3.8)$$

where **F** is called the fundamental matrix for the two views under consideration. As the points **x′** lies on the epipolar line $\mathbf{l}'$,

$$\mathbf{x}'^{\mathbf{T}}\,\mathbf{l}' = 0. \tag{3.9}$$

Therefore the following equation is satisfied for all pairs of corresponding points:

$$\mathbf{x}'\mathbf{F}\mathbf{x} = 0. \tag{3.10}$$

The importance of the relation (3.10) is that it enables **F** to be computed from image correspondences alone, without the knowledge of the camera parameters or 3D structure [1].

3.3 Projective and metric reconstruction

In order to determine the 3D reconstruction, we need to calculate the camera projection matrices first. For this purpose, we will use the knowledge about two-view geometry represented by the fundamental matrix.

In fact that the fundamental matrix **F** only depends on the image coordinate frames (image correspondences). It does not depend on the choice of world coordinate frame and is unchanged by a projective transformation of the 3D space. If we denote a 4x4 matrix representing a projective transformation in 3D as **H**, then the fundamental matrices corresponding to the pair of camera matrices (**P**, **P′**) and (**PH**, **P′H**) are the same. In fact, we can find the whole family of projection matrices differing by affine transformations corresponding to the same fundamental matrix **F.** The opposite is not true and a pair of camera matrices uniquely determines the fundamental matrix **F** [1].

According to the previous paragraph, two images can be used to determine a pair of camera projection matrices, but not uniquely. For this purpose, the world frame can be chosen to be the same as the coordinate frame of the first camera and the second camera is then determined by the two-view geometry. The pair of camera matrices must satisfy following formulas:

$$\mathbf{P} = [\mathbf{I} \mid \mathbf{0}] \tag{3.11}$$

$$\mathbf{P}' = [[\mathbf{e}']_\times \mathbf{F} + \mathbf{e}'\mathbf{v}^{\mathbf{T}} \mid \lambda \mathbf{e}'] \;. \tag{3.12}$$

This equation is not completely determined by the two-view geometry, but has 4 more degrees of freedom given by vector $\mathbf{v} \in \mathrm{R}^3$ and a scalar value λ. Vector **v** determines the position of the reference plane and scalar λ specifies the global scale of the reconstruction. However, missing knowledge of parameter values **v** and λ prevents us from solving the equation (3.12) uniquely and identifying the pair of camera projection matrices.

So called projective reconstruction approach enables to determine a camera projection pair using relations (3.11), (3.12) by setting the unknown parameters **v** and λ as arbitrary values. As a result, the camera pair computed in this manner describes only the world under a projective transformation that is the original world scene perceived as a

Euclidean 3D space is "deformed" by the projective transformation of the 3D space. The effect of the projective transformation on the cube is shown in Fig. 3.5.

This approach can be used but requires computing projective reconstruction first and then upgrading it to a metric one using one of the methods available. However, upgrading the projective reconstruction to a metric one can be a very difficult problem in case the image correspondences are not sufficiently accurate, or if the modeled 3D structure or camera trajectory has no special properties to facilitate the identification of the upgrading transformation [1].

It is possible to determine the camera projection matrices leading directly to metric reconstruction (see Fig. 3.5) if we know the camera internal parameters. The knowledge of camera internal parameters allows us to specialize the fundamental matrix into a so termed essential matrix [1].

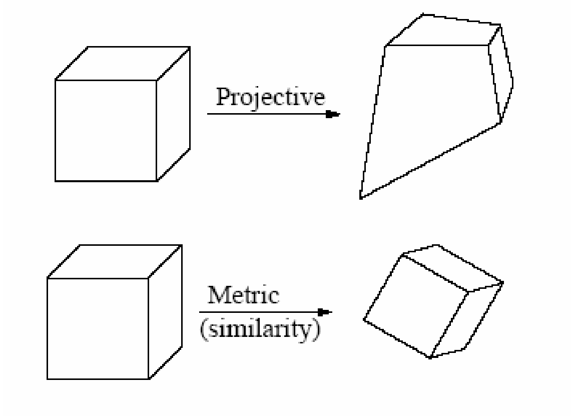


Fig. 3.5. A cube under projective and metric transformation.

The essential matrix is a calibrated form of the fundamental matrix and the relation between the fundamental and the essential matrix is defined as

$$\mathbf{E} = \mathbf{K}^{\mathrm{T}}\mathbf{F}\mathbf{K} \tag{3.13}$$

in case the calibration matrix **K** is the same for both views of the image pair. This formula can also be written in the form:

$$\mathbf{E} = [\mathbf{t}]_{\mathrm{x}}\mathbf{R}, \tag{3.14}$$

where $[\mathbf{t}]_{\mathrm{x}}$ denotes the skew-symmetric matrix[3]

$$[\mathbf{t}]_{\mathrm{x}} = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}. \tag{3.15}$$

[3] Skew-symmetric, or antisymmetric matrix, is a square matrix, the transpose of which is also its negative.

The essential matrix has only five degrees of freedom and this fact allows us to retrieve the camera matrices from **E** up to scale and four-fold ambiguity. This is the important difference between the fundamental and the essential matrix: while the fundamental matrix defines the cameras up to the projective ambiguity and overall scale, the essential matrix determines four possible solutions for the cameras, except for the scale factor, hence the process of upgrading the projective structure to metric one is not necessary. In this work, the direct metric reconstruction approach is applied.

3.4 Retrieving camera matrices from the essential matrix

A pair of camera matrices can be uniquely determined from the corresponding essential matrix [1]. For this purpose, we need to define so called normalized camera matrix, which is the camera matrix having the identity matrix as a calibration matrix. This normalized camera matrix, denoted $\mathbf{P_e}$**,** describes only the motion parameters since the effect of the calibration matrix has been removed:

$$\mathbf{P_e} = \mathbf{K}^{-1}\mathbf{P} = [\mathbf{R} \,|\, \mathbf{t}]. \tag{3.16}$$

In order to find $\mathbf{P_e}$, we need to factorize **E** using the SVD as

$$\mathbf{E} = \mathbf{U}\mathrm{diag}(1,1,0)\mathbf{V}^{\mathrm{T}} \tag{3.17}$$

with **U** and **V** chosen such that their determinant is positive. Without the loss of generality we may assume that the first camera matrix is $\mathbf{P_e} = [\mathbf{I}|\mathbf{0}]$**.** Then four possible choices for the second camera matrix $\mathbf{P'_e}$ are:

$$\mathbf{P'_{e1}} = [\mathbf{UWV}^{\mathrm{T}} | +\mathbf{u}_3] \tag{3.18}$$

$$\mathbf{P'_{e2}} = [\mathbf{UWV}^{\mathrm{T}} | -\mathbf{u}_3] \tag{3.19}$$

$$\mathbf{P'_{e3}} = [\mathbf{UW}^{\mathrm{T}}\mathbf{V}^{\mathrm{T}} | +\mathbf{u}_3] \tag{3.20}$$

$$\mathbf{P'_{e4}} = [\mathbf{UW}^{\mathrm{T}}\mathbf{V}^{\mathrm{T}} | -\mathbf{u}_3] \tag{3.21}$$

where $\mathbf{u}_3 = \mathbf{t}$ is the third column of the matrix **U**, and **W** is defined as

$$\mathbf{W} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{3.22}$$

In order to determine which of the four solutions corresponds to the true configuration, one point is theoretically sufficient to resolve the correct camera pair: the image point is successively triangulated using all possible camera pairs to yield the 3D space coordinates of the point $\mathbf{X_i} = [\mathbf{X_{i1}}\ \mathbf{X_{i2}}\ \mathbf{X_{i3}}\ \mathbf{X_{i4}}]^{\mathbf{T}}$ and each time the coefficients

$$c_1 = \mathbf{X}_{\mathrm{i3}}\mathbf{X}_{\mathrm{i4}}, \quad c_2 = (\mathbf{P'_{ei}}\,\mathbf{X_i})_3\,\mathbf{X}_{\mathrm{i4}} \tag{3.23}$$

are calculated. In case they are both positive (that means $\mathbf{X}_i$ is located in front of cameras), then $\mathbf{P}'_{ei}$ is the correct solution for the second camera.

3.5 Linear triangulation method

The process of acquiring 3D points of the reconstructed scene from image correspondences is called a linear triangulation method. In case we have calculated at least two camera projection matrices, we can compute 3D points of the reconstructed scene corresponding to defined image correspondences. The procedure is as follows:

Given camera matrices **P, P′** and a pair of image correspondences $\mathbf{x}=(x\ y\ w)^{\mathrm{T}}$ and $\mathbf{x}'=(x'\ \ y'\ w')^{\mathrm{T}}$, a 3D point $\mathbf{X}$ is described using the relation (3.6), such that

$$\mathbf{x} = \mathbf{P}\mathbf{X} \tag{3.24}$$
$$\mathbf{x}' = \mathbf{P}'\mathbf{X}. \tag{3.25}$$

These equations allows us to calculate the 3D position for each feature pair using the linear triangulation method, which combines the equations from (3.24) and (3.25) into a form **AX** = 0 with **A** composed as

$$\mathbf{A} = \begin{bmatrix} x\ \mathbf{p}^3 - \mathbf{p}^1 \\ y\ \mathbf{p}^3 - \mathbf{p}^2 \\ x'\ \mathbf{p}'^3 - \mathbf{p}'^1 \\ y'\ \mathbf{p}'^3 - \mathbf{p}'^2 \end{bmatrix}, \tag{3.26}$$

where $\mathbf{p}^i, \mathbf{p}'^i$ are corresponding rows of **P**, **P′** for $i \in \{1,2,3\}$. The least square solution of the system, defining the coordinates of a 3D point **X**, can be found for example using singular value decomposition (SVD) [1] [17], see Section 4.1.

The accuracy of reconstructed 3D points should be analyzed by a reprojection error determined as

$$\sigma = \|\mathbf{x} - \hat{\mathbf{x}}\|, \tag{3.27}$$

$\hat{\mathbf{x}}$ is the reprojected position of image point **x** calculated from equation $\hat{\mathbf{x}} = \mathbf{X}\mathbf{P}$.

It can be convenient to represent 3D points by their depths sometimes. The term depth of points is used to define the distance of 3D points from the camera centre **C** in the direction of its principal ray, see Fig. 3.6. Let us consider the camera matrix $\mathbf{P} = [\mathbf{M} \mid \mathbf{p}_4]$, where **M** is a 3 x 3 matrix, which projects a space point $\mathbf{X} = (X, Y, Z, T)^{\mathrm{T}}$ to the image point $\mathbf{x} = w\ (x, y, 1)^{\mathrm{T}}$. Then the depth of the point **X** in front of the camera **P** is defined as:

$$\text{depth}\ (\mathbf{X}; \mathbf{P}) = \frac{\operatorname{sign}(\det(\mathbf{M}))\, w}{T \left\| \mathbf{m}^3 \right\|}, \tag{3.28}$$

where vector $\mathbf{m}^3$ constitutes the camera's principal ray, which corresponds to the third row of **M** [1].[4]

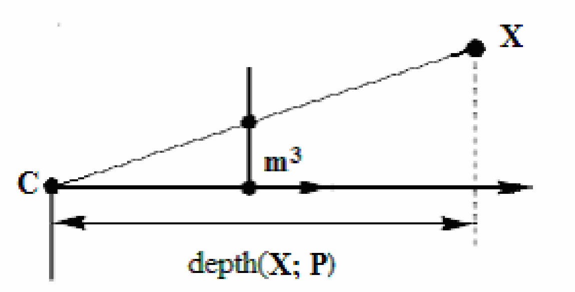


Fig. 3.6. Demonstration of the depth of the space point **X** from the camera **P** with the center **C** and the principal ray $\mathbf{m}^3$.

[4] det(**M**) is a notation for determinant of the matrix **M**. sign is a notation for the signum function.

4 Computation of the fundamental matrix

As was shown in Chapter 3, the fundamental matrix determines the relationship between two projections of one space point taken with two different cameras through the mapping from points to lines. In this chapter, a robust 8-point algorithm to estimate the fundamental matrix is presented. This approach, adopted from [1], is used within the method suggested in this work.

The fundamental matrix is a singular 3x3 matrix determined up to the scale, it has rank 2 and its right and left null-space correspond to the epipoles. Since the fundamental matrix has seven degrees of freedom [1], having at least 7 point correspondences $\mathbf{x_i} \leftrightarrow \mathbf{x_i}'$, equation (3.10) can be used to compute the unknown matrix **F**.

4.1 The normalized 8-point algorithm

The simplest way to compute the fundamental matrix consists in solving a set of linear equations for 8 or more correspondence pairs. The equation (3.10) can be rewritten in the following form:

$$[\; xx' \;\; yx' \;\; x' \;\; xy' \;\; yy' \;\; y' \;\; x \;\; y \;\; 1 \;]\,\mathbf{f} = 0 \tag{4.1}$$

with $\mathbf{x}$=($x\ y\ w)^{\mathrm{T}}$**,** $\mathbf{x}'$=($x'\ y'\ w')^{\mathrm{T}}$ representing corresponding image point in homogenous coordinates, w =1 and $\mathbf{f} = (\mathrm{F}_{11}\ \mathrm{F}_{12}\ \mathrm{F}_{13}\ \mathrm{F}_{21}\ \mathrm{F}_{22}\ \mathrm{F}_{23}\ \mathrm{F}_{31}\ \mathrm{F}_{32}\ \mathrm{F}_{33})^{\mathrm{T}}$. From a set of n point matches, we obtain a set of homogenous equations:

$$\mathbf{Af} = \begin{bmatrix} x_1x_1' & y_1x_1' & x_1' & x_1y_1' & y_1y_1' & y_1' & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_nx_n' & y_nx_n' & x_n' & x_ny_n' & y_ny_n' & y_n' & x_n & y_n & 1 \end{bmatrix} \mathbf{f} = 0 \tag{4.2}$$

If the correspondences were exact, the matrix **A** would have rank 8 and a unique solution up to scale would exist. However, because of the presence of noise, **A** has rank 9 and the solution needs to be obtained in a least-squares sense, for example using singular value decomposition (SVD) [1] [17]. Applying SVD to **A** yields the decomposition

$$\mathbf{A} = \mathbf{U}\,\mathbf{D}\,\mathbf{V}^{\mathrm{T}}, \tag{4.3}$$

where **U** and **V** are orthogonal matrices and **D** is a diagonal matrix with non-negative entries. The least square solution for **f** using $n \geq 8$ image point correspondences is a singular vector corresponding to the smallest singular value of **A**, which is the last column of **V** [1]**.** The matrix **F** found like this does not have rank 2 in the presence of the noise. As a result, epipolar lines do not meet in a common epipole. Therefore **F** needs to be corrected to enforce rank-2 constraint [1].

To succeed with the above described 8-point method, normalization of the image point correspondences before solving the equation (4.2) is required. The need to apply normalization arises from the fact that the mapping represented by the fundamental matrix is not invariant to similarity transformations of the image. The normalization suggested in [1] consists in a translation and scaling of each image point to that their centroid is at the origin of the coordinates and the average distance of the points from the origin is equal to $\sqrt{2}$.

I applied a normalized 8-point algorithm for computing **F,** which is used as an initial **F** estimation. The algorithm is easy to implement and it can provide good results if proper normalization of input data is performed [1]. The implementation of the normalized 8-point algorithm is summarized in Algorithm 4.1.

Objective:

Given $n \geq 8$ image point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the fundamental matrix **F** such that $\mathbf{x}'^{\mathrm{T}}_i \mathbf{F} \mathbf{x}_i = 0$.

Algorithm:

(a) **Normalization***:* Transform the image coordinates according to $\hat{\mathbf{x}}_i = \mathbf{T}\mathbf{x}_i$ and $\hat{\mathbf{x}}'_i = \mathbf{T}'\mathbf{x}'_i$, where **T** and **T′** are normalizing transformations including a translation and scaling of each image point so that their centroid is at the origin of the coordinates and their average distance from the origin is equal to $\sqrt{2}$.

(b) Find the fundamental matrix $\hat{\mathbf{F}}'$ corresponding to the matches $\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i$ by

- **Linear solution:** Determine $\hat{\mathbf{F}}$ from $\hat{\mathbf{A}}\hat{\mathbf{f}} = 0$ applying the SVD to $\hat{\mathbf{A}}$, which is composed of the matches $\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i$.
- **Constraint enforcement:** Replace $\hat{\mathbf{F}}$ by $\hat{\mathbf{F}}'$ such that the determinant of $\hat{\mathbf{F}}' = 0$. Use the SVD to get $\hat{\mathbf{F}} = \mathbf{U}\mathbf{D}\mathbf{V}^{\mathrm{T}}$, where $\mathbf{D} = diag(r,s,t)$[5] with $r \geq s \geq t$. Then $\hat{\mathbf{F}}' = \mathbf{U}\, diag(r,s,0)\, \mathbf{V}^{\mathrm{T}}$.

(c) **Denormalization:** Set $\mathbf{F} = \mathbf{T}'^{\mathrm{T}}\hat{\mathbf{F}}'\mathbf{T}$, where matrix **F** is the fundamental matrix corresponding to the original data $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$.

Algorithm 4.1. The normalized 8-point algorithm for estimating the fundamental matrix [1].

4.2 Robust algorithm

The previous approach for estimating **F** performs sufficiently well if we work with correct correspondences only. However, for real scene images the tracking techniques described in Section 2 also sometimes return pairs of feature points which although very well correlating do not correspond to the same point in the scene. Therefore the set of obtained putative matches is contaminated with some subset of wrong matches, so called outliers. Similarly, good correspondences are called inliers. Even a small set of outliers may cause the result to be unusable hence the outliers need to be removed to acquire correct problem solution.

[5] *diag*(*r*,*s*,*t*) is a notation for a diagonal matrix with values *r*, *s*, *t* on the main diagonal.

A solution for this problem can be obtained by using the RANSAC (RANdom SAmple Consensus) algorithm [1], which divides the data set into inliers and outliers and uses the inliers for the robust solution estimation of the two-view geometry [1]. The RANSAC idea is based on repeating a procedure involving random selection of a subset of matches and evaluating a solution from them. The correct solution is identified as the solution with the largest number of inliers, which are determined considering how closely a correspondence pair satisfies the epipolar geometry. The first-order approximation of the geometric error known as Sampson distance is supposed to give good results in this case [1]. The Sampson distance *d* is defined by (4.4).

$$d = \frac{\left(\mathbf{x'}^{\mathrm{T}}\,\mathbf{F}\,\mathbf{x}\right)^2}{\left(\mathbf{Fx}\right)_1^2 + \left(\mathbf{Fx}\right)_2^2 + \left(\mathbf{F}^{\mathrm{T}}\mathbf{x}'\right)_1^2 + \left(\mathbf{F}^{\mathrm{T}}\mathbf{x}'\right)_2^2}\ , \tag{4.4}$$

where $(\mathbf{Fx})_{\mathrm{j}}^2$ represents the square of the j-th entry of the vector $\mathbf{Fx}$.

Objective:

Given a set of image putative matches $\{\mathbf{x}_{\mathrm{i}} \leftrightarrow \mathbf{x}'_{\mathrm{i}}\}$, determine the fundamental matrix **F** between two views using RANSAC.

Algorithm:

```
$N$ = inf, trial_count = 0, max_ trials=1500
While ($N$ > trial_count && trial_count ≤ max_trials) repeat
    • Select a random sample of $s$ = 8 correspondences and compute the fundamental matrix F using the normalized 8-point algorithm.
    • Calculate the Sampson distance $d$ for each putative correspondence, using the equation (4.4)
    • Compute the number of inliers consistent with F by the number of correspondences for which abs($d$) < $t$, where $t$ is threshold distance for considering a point to be an inlier. The value $t$ was set to 0.5 pixel.
    • Update the estimate of $N$ using constraint (4.5). Increment the trial_count by 1.
Choose the F with the largest number of inliers and refine it based on all inliers by solving equation 4.2 for all inliers.
```

Algorithm 4.2. Algorithm for estimating **F** between two views using RANSAC.

Since it is computationally infeasible to try every possible sample, we will only examine a finite number. The number of RANSAC iterations is chosen to ensure with a certain probability that at least one of the random selected samples, consisting of *s* points, is free of outliers. The probability is usually set to 0.99. The following relation defines the number of RANSAC iterations *N* required to obtain a correct solution with probability $p = 0.99$.

$$N = \log(\ 1 - p\ )\ /\ \log(\ 1 - (\ 1 - \varepsilon\ )^{s}\ ), \tag{4.5}$$

s is a given size of a sample, ε is an estimate of the outlier ratio estimated as follows:

$$\varepsilon = 1\text{-(number of inliers)/(total number of points).} \qquad (4.6)$$

The overview of the two-view geometry computation algorithm using RANSAC applied in this work is given in Algorithm 4.2 [1].

The importance of the presented algorithm consists not only in the estimation of two-view geometry but also in the classification of the inlying correspondences suited for tracking and structure computation.

4.3 Experiments on colonoscopic data

In order to compare epipolar geometry results for both tracking methods, the following experiment was applied. A part of the input image sequence consisting of 7 successive images was used. 6 image pairs were established over this sequence such that the first image was paired with every other image in the sequence in order to create image pairs with different displacement between corresponding views (so the particular image pairs were established between images 1-2, 1-3, 1-4, 1-5, 1-6 and 1-7, where numbers correspond to the order of views in the sequence). Algorithm 4.2 was applied 30 times to every image pair and the mean reprojection error (see equation 3.27) was evaluated. This process was performed using both tracking methods applied in this work. The results illustrating reprojection errors calculated in the latter images within pair are shown in Fig. 4.1. Reprojection errors obtained in primary images within pair were almost zero values. The results show that the mean reprojection errors are very similar for both tracking methods and the error increases depending on displacement of views.

In the next experiment, a mean reprojection error using different values *t* of threshold distance for classifying inliers was estimated. The results are given in Fig. 4.2. The same set of image pairs as in the previous experiment was used and results were evaluated over 30 trials per every image pair as well. It can be seen that the quantity of reprojection error depends on the threshold value. However, the lower the threshold value is, the fewer inlying correspondences we obtain. In the algorithm implemented in this work, the used threshold value for deciding inliers is set to *t*=0.5, since it seems to be a reasonable compromise solution to obtain a satisfactory number of correspondences (basically about 500-1000 between successive images) and avoid large errors.

An application of Algorithm 4.2 is demonstrated in Fig. 4.3. It can be seen that there are still several mishmashes remaining although the experiments showed that the reprojection error is relatively small and epipolar geometry should be evaluated basically correctly. However, a small reprojection error not always guarantees a small error of 3D points, this depends on relative position of the cameras and location of space points.

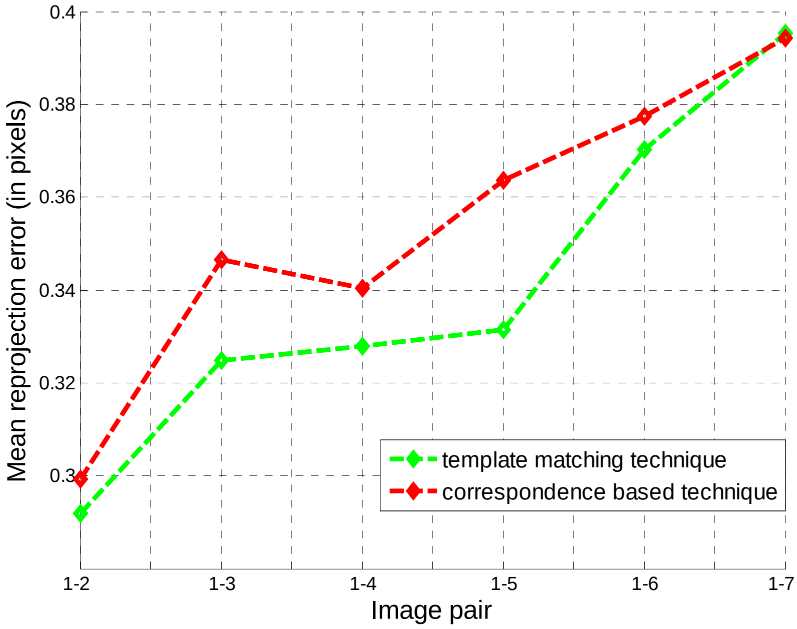


Fig. 4.1. Mean reprojection error calculated for image pairs with different displacement between corresponding images. Epipolar geometry was estimated using Algorithm 4.2. over 30 trials per every image pair.

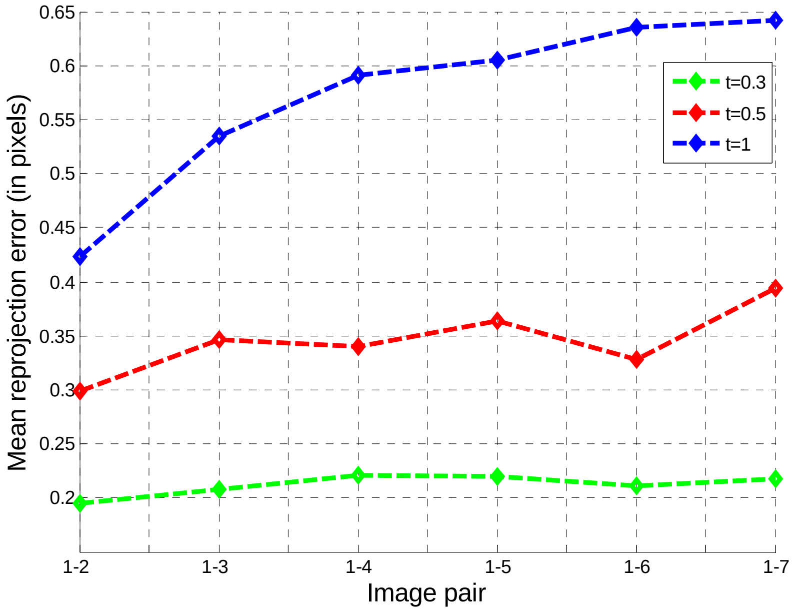
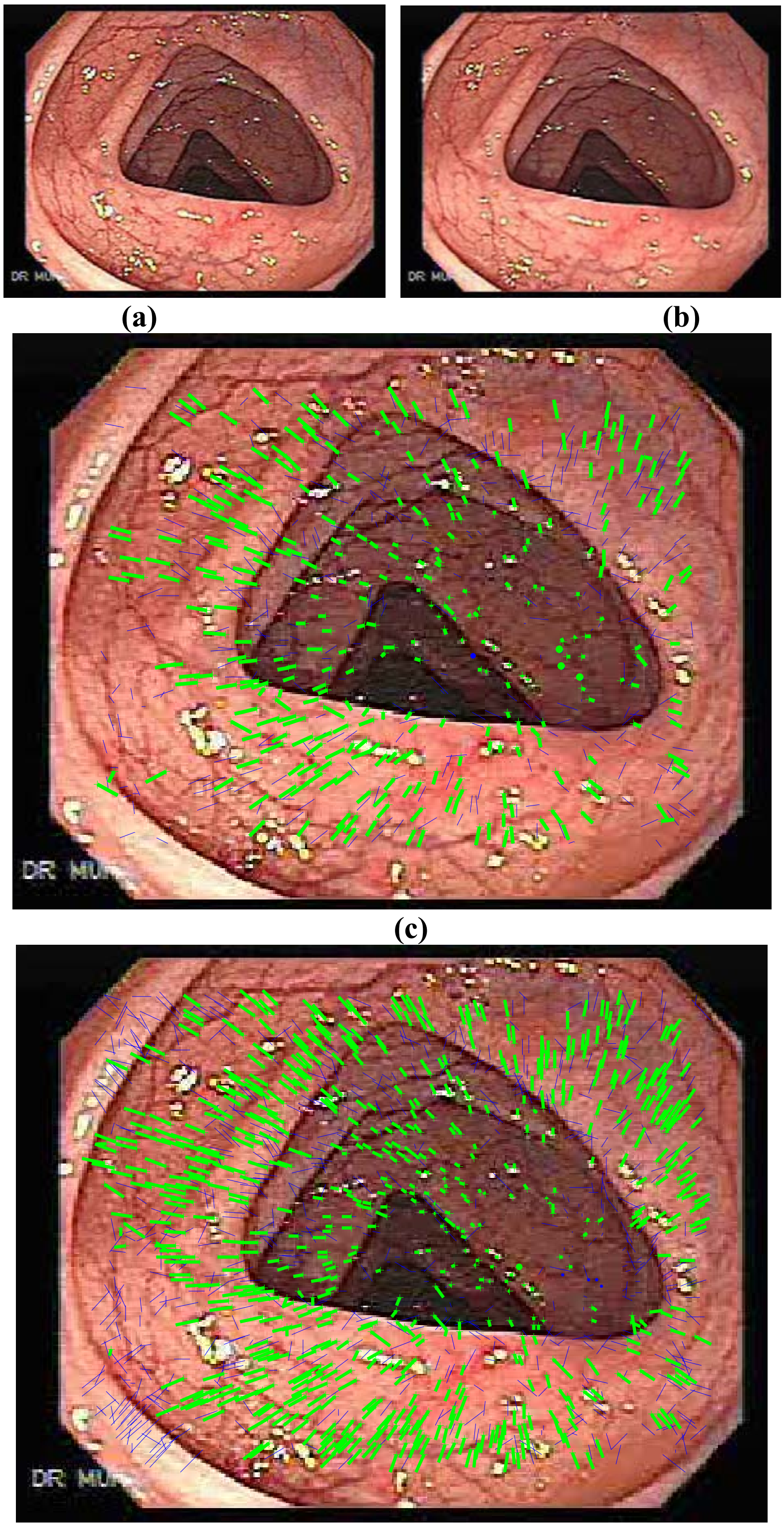


Fig. 4.2. Mean reprojection error calculated for image pairs with different displacement between corresponding images. Feature tracking was performed by correspondence based technique and epipolar geometry was estimated using Algorithm 4.2 over 30 trials per every image pair. The comparison of results obtained by using three different values *t* of threshold distance for classifying inliers is given.

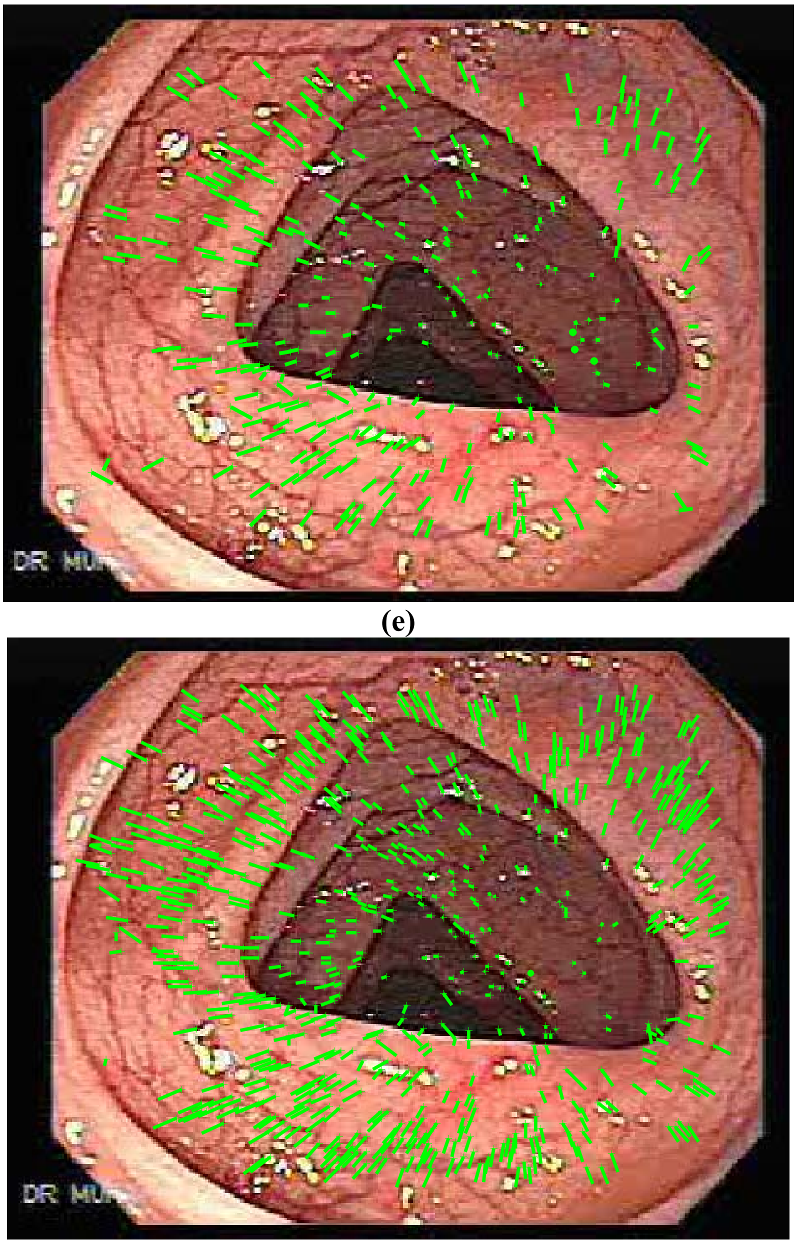


(a) **(b)**

(c)

(d)

Fig. 4.3: Classification of the inlying correspondences using the robust normalized 8-point algorithm to estimate **F**.



(e)

(f)

Fig. 4.3: Classification of the inlying correspondences using the robust normalized 8-point algorithm to estimate **F**. (a) (b) Input image pair. (c) (d) Putative matches superimposed on the left image. Green lines represent a subset of putative matches which were identified as inliers; blue lines correspond to matches classified as outliers (mismatches). Correspondence-based technique to obtain putative matches was applied in (c), while putative matches acquired by template matching technique are given in (d). (e) (f) Final inliers consistent with the estimated **F** corresponding to (c) (d). A threshold distance for considering a point to be an inlier was set as t=0.5.

5 Auto-calibration

In order to achieve metric reconstruction we need to first identify camera internal parameters. Recovering the unknown intrinsic parameters of the cameras is called calibration. Unlike a conventional calibration where the camera parameters are determined using special calibration objects [1] or properties of the scene (for instance vanishing points of orthogonal directions [1]), auto-calibration is a process of computing calibration matrix directly from uncalibrated images using constraints on camera parameters. In this chapter, the auto-calibration problem described and a solution for colonoscopic camera auto-calibration is presented.

5.1 Overview of auto-calibration methods

Two main types of auto-calibration approaches are described in literature, stratified auto-calibration methods and direct auto-calibration methods. Stratified auto-calibration methods are based on upgrading a projective reconstruction to an affine one, and after that the affine-to-metric transformation is performed to achieve a metric reconstruction [20, 21]. Direct auto-calibration methods, on the other hand, directly compute the projective-to-metric upgrading homography.

The direct methods include the approach based on estimating the absolute dual quadric, encapsulating the plane at infinity and camera intrinsic parameters, or the method using the Kruppa equations, which is historically the first auto-calibration method ever used [18, 19]. The Kruppa equations are two independent quadratic equations arising from the fundamental matrix, which generally defines seven two-view constraints. Five of them determine the camera motion and two more specify the camera intrinsics – these are the Kruppa equations. The advantage of this auto-calibration technique is that it works only with pairwise epipolar geometry and no consistent projective reconstruction is necessary. Recently, auto-calibration methods solving two-view focal-length estimation problem have been discovered and used. They utilize the techniques of algebraic geometry such as the Gröbner basis or the hidden variable technique [29, 30].

An important aspect of the auto-calibration problem is the problem of critical motion sequences. In some cases, the motion of the camera is not general enough and, as a result, the auto-calibration process has ambiguous solutions. These critical configurations, also termed degenerate configurations, differ depending on calibration parameters (the number of known or fixed parameters) and the number of views [23, 24, 25, 26, 28].

5.1.1 Constant focal length auto-calibration

We shall assume that the endoscopic camera has a fixed focal length, square pixels, a principal point in the image centre and a zero skew. The only unknown variable from the intrinsic parameters we need to determine is the focal length. As mentioned in the previous chapter, once we have all the internal parameters it is possible to estimate the essential matrix and, subsequently, to calculate the camera matrices and to reconstruct the points **X**.

Estimation of the constant focal-length from two semi-calibrated views (semi-calibrated or partially calibrated means that all camera intrinsics are known except the fixed focal length) is possible through the fundamental matrix [27, 28]. Generally, if two camera views are uncalibrated, seven points are the minimal requirement to compute the fundamental matrix (since the fundamental matrix has seven degrees of freedom). In the semi-calibrated case, it is possible to recover the essential matrix and thus the unknown focal-length from six corresponding points. The fully-calibrated views require only five points to estimate the essential matrix.

Stewénius et al have proposed an algorithm to solve the 6-point focal length problem [29]. They have applied a special powerful mathematical tool for handling a polynomial system called the Gröbner basis technique [31,32,33]. They have shown that there are up to 15 solutions to the six-point algorithm. Moving from non-minimal solutions (eight and seven point techniques) to minimal algorithm provides some benefits. The six-point algorithm has fewer degenerate configurations than the seven-point algorithm [34] and, moreover, experiments have demonstrated that the six-point algorithm sometimes offers better performance for focal length estimation than the seven-point algorithm [29].

In 2006, Hongdong Li provided an alternative solution to the 6-point focal-length problem [30], simpler in comparison with the one originally proposed in [29]. This algorithm is based on a hidden-variable technique [33] instead of the Gröbner basis technique and had been tested on both synthetic and real images (with different levels of noise). Satisfactory results were obtained for both cases [30]. He also conducted experiments comparing the results achieved by his algorithm and the method [29], and reported that no significant difference was found.

The critical motion sequences for the auto-calibration of a constant focal length from two views were classified by Sturm et al in [28]. These critical (degenerate) configurations constitute the cases when the optical axes are parallel to each other, or when the optical axes intersect at a finite point which is equidistant from the optical centers. Degenerate configurations carry theoretical singularities, also termed generic singularities, resulting in the fact that the problem has no or incorrect solution. These generic singularities cannot be overcome by any algorithm. In contrast, there are also artificial (also termed non-generic) singularities, which depend on the applied algorithm. Some algorithms only have generic singularities, such as [30, 29].

5.2 Estimation of the colonoscopic camera focal-length

In order to estimate the focal-length parameter of the camera, which was used to take the input image colonoscopic sequence, the two-view six-point algorithm provided by Hondong Li in [30] was applied. As was mentioned above, this algorithm utilizes the hidden-variable technique suitable for elimination of variables from polynomial equation system. The theoretical background of the algorithm, the description of implementation and experimental results for the colonoscopic data are given in the following sections.

5.2.1 Solution for the six-point focal-length problem by Hongdong Li

Outline of the theoretical background of this six-point technique adopted from [30] is basically as follows. Starting with the knowledge of the essential matrix properties, constraint (4.8) can be rewritten into the form

$$\mathbf{F} = \mathbf{K}^{\mathbf{-T}}\mathbf{E}\mathbf{K}^{\mathbf{-1}}. \tag{5.1}$$

The essential matrix has rank 2, as well as the fundamental matrix, but in addition it has a property that the two non-zero singular values of the essential matrix are equal. This property leads to the important cubic constraint on the essential matrix, adapted from [38]:

$$\mathbf{2EE^{T}E}\ \mathbf{-}\ \mathrm{trace}\,\mathbf{(EE^{T})E} = 0\,, \tag{5.2}$$

where trace is a notation for the algebraic operation of summing the elements on the main matrix diagonal.

Since the six-point algorithm works with semi-calibrated cameras, we are allowed to assume the calibration matrix without loss of generality in the form

$$\mathbf{K} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{5.3}$$

For a reason which will soon become clear, this matrix can be rewritten using $w = f^{-2}$ as

$$\mathbf{Q} = w^{-1}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & w \end{bmatrix}. \tag{5.4}$$

As was defined previously, the epipolar constraint (3.10) is defined in the form

$$\mathbf{x'}\,\mathbf{Fx} = 0\,. \tag{5.5}$$

It can be written analogous to (4.2) as

$$\mathbf{\widetilde{x}^{T}\widetilde{F}} = 0\,, \tag{5.6}$$

where

$$\mathbf{\widetilde{x}} = [\,\mathbf{x_1x_1'}\ \ \mathbf{x_2x_1'}\ \ \mathbf{x_3x_1'}\ \ \mathbf{x_1x_2'}\ \ \mathbf{x_2x_2'}\ \ \mathbf{x_3x_2'}\ \ \mathbf{x_1x_3'}\ \ \mathbf{x_2x_3'}\ \ \mathbf{x_3x_3'}\,]^{\mathbf{T}} \tag{5.7}$$

$$\mathbf{\widetilde{F}} = [\,\mathbf{F_{11}}\ \ \mathbf{F_{12}}\ \ \mathbf{F_{13}}\ \ \mathbf{F_{21}}\ \ \mathbf{F_{22}}\ \ \mathbf{F_{23}}\ \ \mathbf{F_{31}}\ \ \mathbf{F_{31}}\ \ \mathbf{F_{33}}\,]^{\mathbf{T}}. \tag{5.8}$$

Using all six point correspondences we get a 6x9 matrix (one row per correspondence). It is possible to derive that the fundamental matrix satisfies the equation

$$\mathbf{F} = x\,\mathbf{F_0} + y\,\mathbf{F_1} + z\,\mathbf{F_2}\,, \tag{5.9}$$

where x, y, z are unknown scalars and $\mathbf{F_0}$, $\mathbf{F_1}$, $\mathbf{F_2}$ represent the basis of the null-space of that 6x9 matrix. Substituting this equation (5.9) into (5.1) and (5.2), the following formula

$$\mathbf{2FQF^TQF} - \mathrm{trace}(\mathbf{FQF^TQ})\mathbf{F} = 0 \tag{5.10}$$

with the unknowns x, y, z, w arises. This constraint gives 9 cubic equations. Putting it together with the singularity condition on the fundamental matrix

$$\det(\mathbf{F}) = 0, \tag{5.11}$$

a system of ten cubic equations in four unknowns is obtained. By resolving this polynomial system the focal-length estimation is completed. This is done by above mentioned hidden variable technique, which works on the basis of solving so called hidden-variable resultant. A more detailed description of this algebra technique is given in [30].

The output of the above described procedure is a set of at most 15 solutions to the six point two-view focal-length problem corresponding to roots of a 15th degree polynomial. Therefore a stage of recovering the truth focal-length from multiple solutions needs to be done.

This problem is solved in paper [30] by procedure based of the following experimental findings. Since the input data are contaminated with noise, the genuine root of polynomial (the genuine focal-length) is not received. Nevertheless, the obtained roots mostly surround the genuine one. Therefore if a sufficiently large number of measurements is used, statistical distribution of all roots displays a peak corresponding to the best root candidate.

The basic idea of the selection process applied in paper [30] is to repeatedly select a random sample of six-point correspondences for estimation of focal-length candidates, consisting of the complex and real roots. Only the real positive ones are kept and subsequently used to estimate the probability density curve using a kernel density estimator defined at point p as

$$\hat{f}_h(p) = \sum_{i=1}^{n} \frac{\kappa(p-p_i)}{h}, \tag{5.12}$$

where $\kappa()$ denotes the kernel function and h the bandwidth. A Gaussian Kernel with fixed bandwidth is used and the bandwidth is set as the expected estimation precision, for instance 1 % of the focal-length.

The author of the paper tested the algorithm on both synthetic data (with various levels of noise and outliers) and real images stating that the method is quite robust to noise and outliers, see Fig. 5.1 for reported results. As well different values of focal-length were tested and it was found that they do not affect the final accuracy. However, the algorithm will fail in degenerate cases for focal-length estimation (for example, when the two optical axes intersect at equal distances, or when the camera underwent a pure translation).

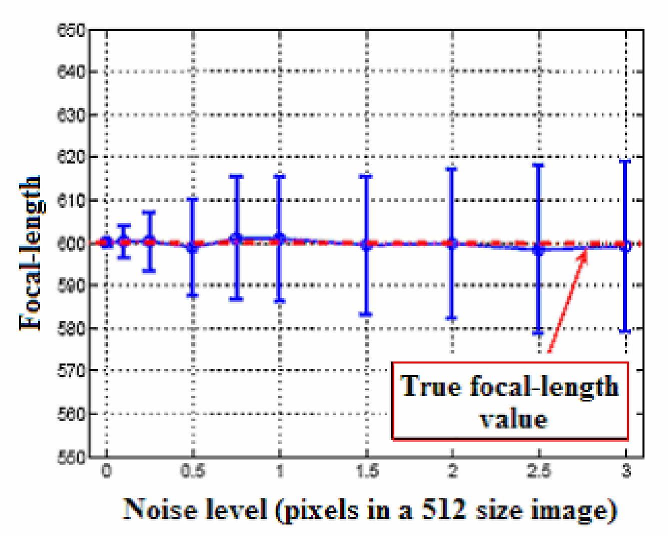


Fig. 5.1. Accuracy in focal length estimation versus noise. Error bars (mean value and standard deviations) of focal-length estimation under different levels of noise. The truth focal-length value is 600.

5.2.2 Implementation of the 6-point method and experimental results

In order to estimate the focal-length of the colonoscopic camera, I use basically the same approach as described in [30]. The main computational process of the applied algorithm is performed by matlab function *SixPtFocal* presented in the appendix of the paper [30]. The input of the function consists of a set of 6 two-view correspondences, the output is a set of computed focal-lengths. As a selection procedure for identifying the best candidate for focal-length, a kernel voting scheme, also suggested in [30], with normal distribution was applied. In order to estimate the focal-length of the colonoscopic camera the following experiments were applied.

The first experiment aimed at testing how many six-point groups of correspondences are needed to be applied to stabilize a peak position of the probability density curve estimated over these groups. Several image pairs were taken and a distribution function was evaluated over 20, 30 and 40 six-point groups. The experiments showed that peak positions of distribution functions are basically very similar when we use various numbers of trials. Therefore 40 six-point groups seem to be enough for focal-length distribution estimation and in further experiments this number of trials was used. Results evaluated for an exemplary image pair are shown in Fig. 5.2.

In the next experiment, a six point method was applied to each pair of successive images in the input colonoscopic image sequence. For each image pair, 40 six-point groups of correspondences were randomly selected and function *SixPtFocal* was applied to them. The probability density curves of the focal-length distributions corresponding to particular image pairs are shown in Fig. 5.3. The graph shows that peak positions of particular distribution function basically vary between 100 and 400. The inaccuracy in results may be caused by the fact that the colonoscopic camera underwent a motion relatively close to a pure translation for witch this six-point method fails.

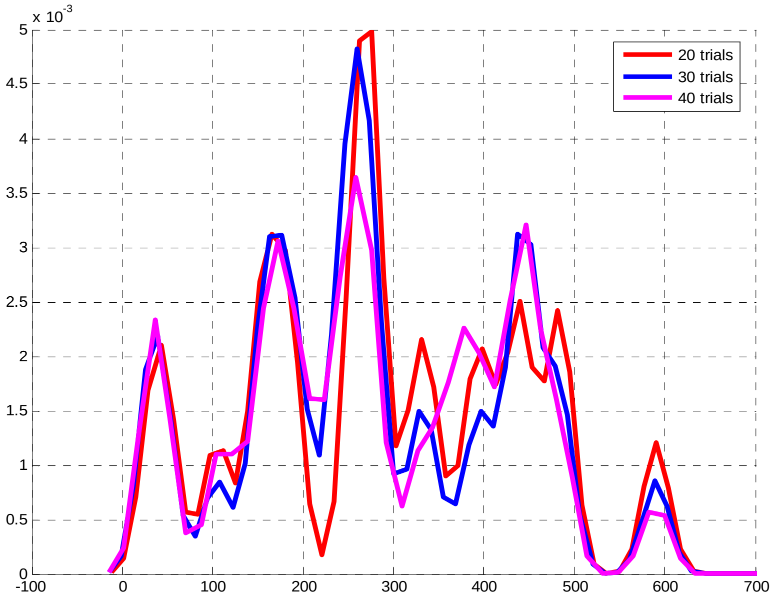


Fig. 5.2. The probability density curves of the focal-length distributions s evaluated for an exemplary image pair over 20, 30 and 40 six-point groups. Bandwidth of the kernel smoothing window was set to 20. Normal distribution is used.

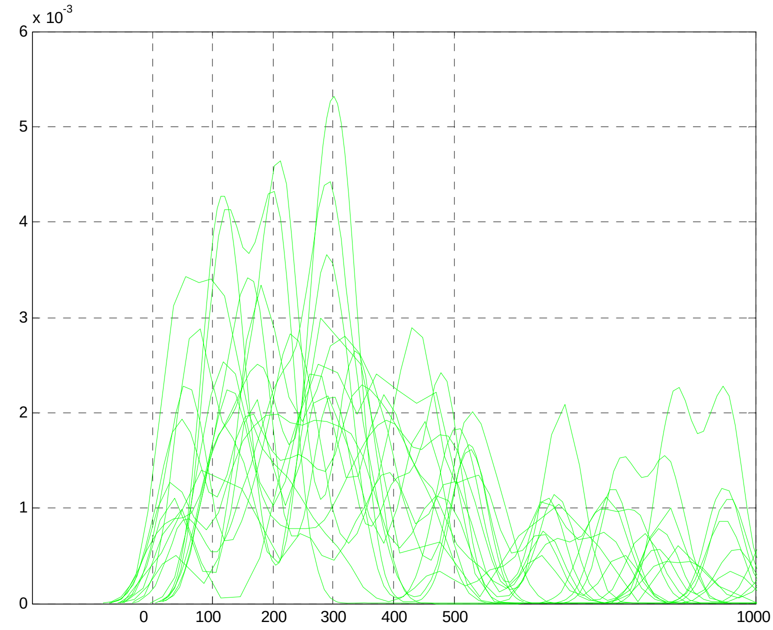


Fig. 5.3. The probability density curves of the focal-length distributions corresponding to particular image pairs in the input colonoscopic sequence. Bandwidth of the kernel smoothing window was set to 20. Normal distribution is used.

I

The focal-length of the colonoscopic camera was estimated as the peak position of the probability density over all roots acquired in this experiment. The resulting density curve of the focal-length distribution for various bandwidths is plotted in Fig. 5.4. The graph shows that the peak position of the distribution function corresponds to values 290, 255 and 247 for values of kernel smoothing window equal to 10, 50 and 100 respectively. Considering the results of this experiment, the focal-length value of colonoscopic camera is supposed to be closed to these values.

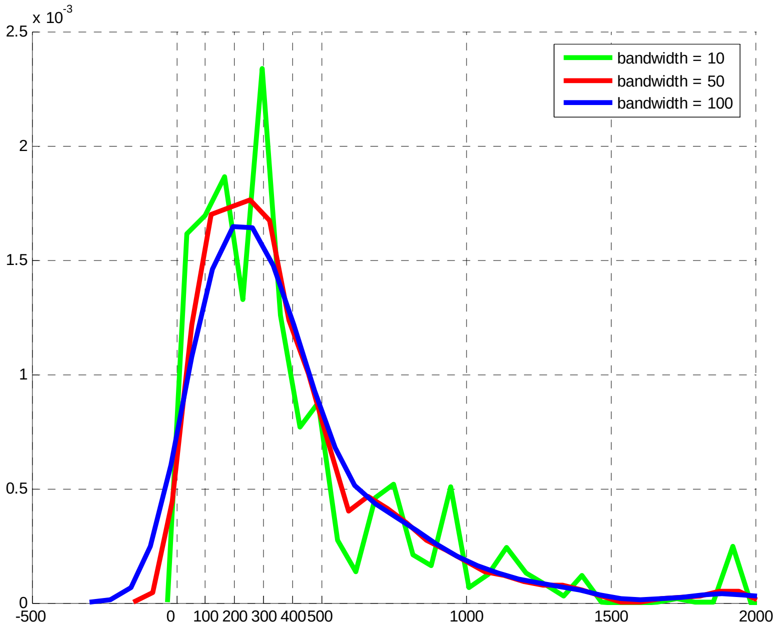


Fig. 5.4. The focal-length estimation for various bandwidths of the kernel smoothing window. The peaks correspond to focal-length values 295 (bandwidth =10), 255 (bandwidth =50) and 247 (bandwidth =100). Normal distribution is used.

6 Multiview metric reconstruction

In this chapter, the solution for a multiview reconstruction based on pair-wise metric reconstructions obtained via calibrated five-point polynomial algorithm is presented.

Once cameras are assumed to be calibrated, we can start to solve the task of estimation of camera matrices. The overall problem of a multiview metric reconstruction can be solved by estimating all pair-wise relative camera rotations and translations (i.e. the rotations and translations are each time estimated for a pair of subsequent views in such a way that the first camera matrix is located at the center of world coordinate frame and the second camera is computed as relative to the first one), from which global camera rotations and translations are subsequently derived.

The relative camera motion between two calibrated views can be estimated using any of the direct relative orientation methods, such as the eight-point, seven-point, six-point or five-point method. As the minimal number of point correspondences required in this case is five, the five-point algorithm is the most suitable with respect to outliers when used with RANSAC [1]. Moreover, the five-point method has the advantage that it works for planar scenes [40]. For these reasons, I use the five-point method to estimate the essential matrix and subsequently the relative camera matrices. In order to achieve robustness of the reconstruction algorithm, it should be used in conjunction with random sample consensus algorithm.

6.1 Pair-wise metric reconstruction using five-point algorithm

The five-point algorithm has been named due to the fact that it can operate on the minimal five point correspondences required to solve the calibrated relative orientation problem. For the first time the five-point problem was explained by Kruppa in 1913 [34] to have at most eleven solutions. This was later improved in [35, 36, 37] stating that there are at most ten solutions corresponding to the roots of a tenth degree polynomial. Although Kruppa also outlined the algebraic method to solve the five-point problem, the method is not feasible using numerical implementation. A more practical solution was devised by Philip [38]. This method has lately been refined by Nistér, first in [39] and then in [40].

In my work I utilize the version of two-view five-point algorithm described in [40]. The Matlab code for this five-point solver, which is based on the powerful Gröbner basis technique [31,32], is available on the internet [41]. The input of the algorithm is a set of five calibrated image pair correspondences and the output is a number of possible solutions (up to ten) for the corresponding essential matrix.

6.1.1 Theoretical background of two-view five-point method

A brief review of five-point method is given below. A more detailed description of the algorithm can be found in [40] and in the bibliography mentioned in [40], because complete understanding of the problematic requires to have specialized knowledge in algebra and algebraic geometry, which is over the scale of this work. The basic ideas, that the five-point method is built on, are to a certain extent analogous to the six-point method presented in the previous chapter.

The five-point algorithm works with calibrated cameras. In this case the image points **x**, **x’** need to be multiplied by the inverse form of calibration matrix **K**:

$$\mathbf{q} = \mathbf{K}^{-1}\mathbf{x} \tag{6.1}$$

$$\mathbf{q}' = \mathbf{K}^{-1}\mathbf{x}' . \tag{6.2}$$

The coplanarity constraint (3.10) is then formulated as

$$\mathbf{x}'\,\mathbf{F}\mathbf{x} = 0 \tag{6.3}$$

$$\mathbf{x}'\,\mathbf{K}^{-T}\mathbf{E}\mathbf{K}^{-1}\mathbf{x} = 0 \tag{6.4}$$

$$\mathbf{q}'\,\mathbf{E}\mathbf{q} = 0 . \tag{6.5}$$

The cubic constraint (5.2) on the essential matrix, defined in previous chapter for the six-point method, is a basis also for the five-point method. We will write it down once more:

$$\mathbf{2EE^TE - trace(EE^T)E} = 0 . \tag{6.6}$$

Each of the five point correspondences has to satisfy equation (6.5), which we shall write analogous to (4.2) and (5.6) as

$$\tilde{\mathbf{q}}^{\mathbf{T}}\tilde{\mathbf{E}} = 0 \tag{6.7}$$

with

$$\tilde{\mathbf{q}} = [\ \mathbf{q_1q_1'}\ \ \mathbf{q_2q_1'}\ \ \mathbf{q_3q_1'}\ \ \mathbf{q_1q_2'}\ \ \mathbf{q_2q_2'}\ \ \mathbf{q_3q_2'}\ \ \mathbf{q_1q_3'}\ \ \mathbf{q_2q_3'}\ \ \mathbf{q_3q_3'}\ ]^{\mathrm{T}} \tag{6.8}$$

$$\tilde{\mathbf{E}} = [\ \mathbf{E_{11}}\ \ \mathbf{E_{12}}\ \ \mathbf{E_{13}}\ \ \mathbf{E_{21}}\ \ \mathbf{E_{22}}\ \ \mathbf{E_{23}}\ \ \mathbf{E_{31}}\ \ \mathbf{E_{31}}\ \ \mathbf{E_{33}}\ ]^{\mathrm{T}} . \tag{6.9}$$

From such five equations we get a 5x9 matrix and by using the SVD an orthogonal basis for the four-dimensional null-space, represented by four vectors, is computed for this matrix. These four vectors can be expressed as four 3x3 matrices $\mathbf{E_1}$, $\mathbf{E_2}$, $\mathbf{E_3}$ and $\mathbf{E_4}$ forming the essential matrix

$$\mathbf{E} = x\mathbf{E_1} + y\mathbf{E_2} + z\mathbf{E_3} + w\mathbf{E_4} \tag{6.10}$$

where *x, y, z, w* are unknown scalars. Since **E** is only defined up to scale, *w* can be set equal to 1 and the number of unknowns in the equation (6.10) is reduced to three. Substituting this equation into the trace constraint (6.6) and considering the fact that the determinant of the fundamental matrix is equal to zero, ten polynomial equations of degree three in the unknowns (*x*, *y*, *z*) are arranged. Next the monomials given in these equations are ordered in GrLex order[6] and defined by a 10x20 matrix. After that the Gröbner basis is computed and used to build a 10x10 action matrix whose eigenvalues and eigenvectors encode the solutions.

[6] In GrLex ordering one first sorts the terms by total degree and then by lexicographical order.

6.1.2 Two-view five-point algorithm together with RANSAC

In practice, the relative orientation problem of recovering the camera matrices needs to be solved in a robust manner to achieve more accurate results. For this purpose, I applied the five-point algorithm within the random sample consensus algorithm [39]. Each time a random sample of five point correspondences is taken and a number of solutions for essential matrix are generated. For each **E**, four possible combinations of **R** and **t** are calculated and true configuration is obtained considering all five correspondences as described in Section 3.4.

Objective:

Given a set of calibrated image correspondences $\mathbf{q_i} \leftrightarrow \mathbf{q'_i}$, determine the relative rotation **R** and relative translation **t** (up to scale) between two views using five-point algorithm together with RANSAC.

Algorithm:

```
$N$ = inf, trial_count = 0, max_ trials=1500
While ($N >$ trial_count && trial_count $\leq$max_trials) repeat
    (a) Select a random sample of $s = 5$ correspondences and compute a number
        of solutions (up to ten) for E using five-point algorithm.
    (b) For each E:
        • Evaluate four possible solutions for the second camera using the
          procedure described in Section 3.4.
        • Triangulate all five correspondences (using relations 3.24, 3.25 and 3.26)
          using all camera pairs and count coefficients according to 3.23. The true
          solution for R and t corresponds to that camera pair for which the
          coefficients $c_1$ and $c_2$ are positive for all image points.
    (c) For all true solutions R, t:
        • Evaluate fundamental matrix $\mathbf{F} = \mathbf{K}^{-T}[\mathbf{t}]_x \mathbf{RK}^{-1}$ and calculate the Sampson
          distance $d$ for each image correspondence $\mathbf{x_i} = \mathbf{Kq_i}$, $\mathbf{x'_i} = \mathbf{Kq'}$ using the
          equation (3.7)
        • Compute the number of inliers consistent with F by the number of
          correspondences for which abs($d$) < $t$, where $t$ is threshold distance in
          pixels for considering a point to be an inlier. Threshold $t$ was set to value
          0.5.
    (d) Select R, t with the largest number of inliers.
    (e) If the solution exists, update estimate of $N$ using constraint (4.5). Increment
        the trial_count by 1. If there is no solution, continue with (a).
Choose the R, t with the largest number of inliers and refine them using all inliers. Reject
outliers from image correspondences.
```

Algorithm 6.1. Algorithm to estimate relative orientation between two views using 5-point algorithm together with RANSAC.

These outgoing configurations are classified by the first-order approximation of the geometric error – the Sampson distance. The correct solution is identified as the solution with the largest number of inliers. Overview of the applied procedure is given in Algorithm 6.1.

6.2 Global rotation and translation estimation

So far we have estimated relative rotations and translations between two arbitrary views. Using these pair-wise relative rotations, global camera rotations can be uniquely derived, since the relative rotations is determined completely. Estimation of global translations is more complicated because in the two-view case relative translations are determined only up to scale. In other words, two views are not enough to determine the scale parameter, but with three views the scale ambiguity can be in general resolved [1].

Assuming three cameras $\mathbf{P_{e1}}$, $\mathbf{P_{e2}}$ and $\mathbf{P_{e3}}$, they can be defined as:

$$\mathbf{P_{e1}} = [\mathbf{I} \mid \mathbf{0}] \tag{6.11}$$
$$\mathbf{P_{e2}} = [\mathbf{R_{21}} \mid \mathbf{t_{21}}] \tag{6.12}$$
$$\mathbf{P_{e3}} = [\mathbf{R_{32}R_{21}} \mid \mathbf{R_{32}t_{21}} + \mathbf{t_{32}}], \tag{6.13}$$

where $\mathbf{R_{21}}$, $\mathbf{t_{21}}$ represent relative rotation and translation between cameras $\mathbf{P_{e1}}$ and $\mathbf{P_{e2}}$, and $\mathbf{R_{32}}$, $\mathbf{t_{32}}$ represent relative rotation and translation between cameras $\mathbf{P_{e2}}$ and $\mathbf{P_{e3}}$. These equations can be rewritten also as

$$\mathbf{P_{e1}} = [\mathbf{I} \mid \mathbf{0}] \tag{6.14}$$
$$\mathbf{P_{e2}} = [\mathbf{R_{21}} \mid \alpha_1 \mathbf{t_{n\,21}}] \tag{6.15}$$
$$\mathbf{P_{e3}} = [\mathbf{R_{32}R_{21}} \mid \mathbf{R_{32}}\, \alpha_1 \mathbf{t_{n\,21}} + \alpha_2 \mathbf{t_{n\,32}}] \tag{6.16}$$

considering the fact that the scale of translation vectors (Euclidian norms of the translations) are not determined uniquely from the applied five-point algorithm and thus we need to specify them using the unknown scalars α_1 and α_2 and vectors $\mathbf{t_{n21}}$ and $\mathbf{t_{n32}}$ having the Euclidian norm equal to 1.

Having an image point $\mathbf{q_i}$ visible in all three views, the equation

$$\beta_i \mathbf{q_i} = \mathbf{P_i X} \tag{6.17}$$

defines the relation among the image point 2D coordinates in the views and its 3D point coordinates. Substituting camera equation (6.14), (6.15) and (6.16), we get

$$\beta_1 \mathbf{q_1} = \mathbf{X} \tag{6.18}$$
$$\beta_2 \mathbf{q_2} = \mathbf{R_{21}X} + \alpha_1 \mathbf{t_{n\,21}} \tag{6.19}$$
$$\beta_3 \mathbf{q_3} = \mathbf{R_{32}R_{21}X} + \mathbf{R_{32}}\, \alpha_1 \mathbf{t_{n\,21}} + \alpha_2 \mathbf{t_{n\,32}} \tag{6.20}$$
$$= \mathbf{R_{32}}(\mathbf{R_{21}X} + \alpha_1 \mathbf{t_{n\,21}}) + \alpha_2 \mathbf{t_{n\,32}} \tag{6.21}$$
$$= \mathbf{R_{32}}\, \beta_2 \mathbf{q_2} + \alpha_2 \mathbf{t_{n\,32}} \tag{6.22}$$

Substituting **X** from the equation (6.18) into the remaining constraints leads to:

$$\beta_2\,\mathbf{q_2} = \beta_1\,\mathbf{R_{21}q_1} + \alpha_1\,\mathbf{t_{n\,21}} \tag{6.23}$$

$$\beta_3\,\mathbf{q_3} = \beta_2\,\mathbf{R_{32}q_2} + \alpha_2\,\mathbf{t_{n\,32}}\,. \tag{6.24}$$

Now we have 6 equations and 5 unknowns - α_1, α_2, β_1, β_2, β_3. The equation system is overdetermined and can be solved by the SVD method.

$$\mathbf{A}\begin{bmatrix}\alpha_1\\ \alpha_2\\ \beta_1\\ \beta_2\\ \beta_3\end{bmatrix} = \mathbf{0} \tag{6.25}$$

In fact, we are only interested in parameters α_1 and α_2, as they determine the correct translation values.

The value of the first constant α_1 can be chosen arbitrarily because the whole 3D reconstruction is determined only up to scale, and hence we can choose a scale factor of the three-dimensional coordinate system we work with. It is comfortable in terms of computation to set α_1 to be equal to 1. Using the equation (6.25), α_2 is calculated and the translation of the third camera $\mathbf{P_{e3}}$ as well as the triplet of camera matrices $\mathbf{P_{e1}}$, $\mathbf{P_{e2}}$ and $\mathbf{P_{e3}}$ can be resolved.

So far we have considered only three camera matrices. Other cameras, or more precisely camera translation vectors, can be calculated in a similar way, using the triplet of subsequent views each time. The constraints (6.14), (6.15) and (6.16) can be used to define a new camera triplet, now formed from the second, the third and the fourth camera. We will suppose that the reference frame of the first camera from this camera triplet is identical with the space coordinate frame, thus we can define cameras in canonical form, as we have done previously. Let us denote the new camera triplet as

$$\mathbf{P'_{e1}} = [\mathbf{I} \mid \mathbf{0}] \tag{6.26}$$

$$\mathbf{P'_{e2}} = \lfloor \mathbf{R'_{21}} \mid \delta_1\,\mathbf{t'_{n\,21}} \rfloor \tag{6.27}$$

$$\mathbf{P'_{e3}} = \lfloor \mathbf{R'_{32}R'_{21}} \mid \mathbf{R'_{32}}\,\delta_1\,\mathbf{t'_{n\,21}} + \delta_2\,\mathbf{t'_{n\,32}} \rfloor, \tag{6.28}$$

From these cameras, we can obtain the information about the ratio δ_2/δ_1, which specifies whether the relative translations between successive cameras are defined in the consistent global scale.

After that the new canonical camera triplet needs to be merged together with the existing cameras and reconstruction as it differs in position of reference plane and also in the scale factor. In fact, only the third camera of the canonical triplet needs to be merged, since the camera pair $\mathbf{P'_{e1}}$ and $\mathbf{P'_{e2}}$ corresponds to the already calculated camera pair $\mathbf{P_{e2}}$ and $\mathbf{P_{e3}}$. Each camera matrix can be defined in terms of the previous camera matrix

Objective:

Given sets of calibrated image correspondences $\mathbf{q}_\mathbf{j}^\mathbf{i}$ through the set of n views, determine corresponding camera matrices $\mathbf{P_j}$ and estimate a sparse structure.

Algorithm

```
i =1
While (i < n-2) repeat
    (1) Select a triplet of successive views and find 3-view point correspondences
        $\mathbf{q}_\mathbf{j}^\mathbf{i} \leftrightarrow \mathbf{q}_\mathbf{j}^\mathbf{i+1} \leftrightarrow \mathbf{q}_\mathbf{j}^\mathbf{i+2}$.
    (2) From correspondences $\mathbf{q}_\mathbf{j}^\mathbf{i} \leftrightarrow \mathbf{q}_\mathbf{j}^\mathbf{i+1}$ and $\mathbf{q}_\mathbf{j}^\mathbf{i+1} \leftrightarrow \mathbf{q}_\mathbf{j}^\mathbf{i+2}$ determine relative rotations
        and relative translations (up to scale) between corresponding views using the
        robust 5-point method described in Algorithm 6.1.
    (3) Perform RANSAC to estimate the consistent scale of relative translations.
        Repeat while $p < 0.99$:
        • Select a random sample of 3-view point correspondences and triangulate
          them to receive a corresponding 3D point.
        • Following the constraints (6.18)-(6.25), calculate parameters determining
          the consistent scale factor of relative translations.
        • Compute canonical camera matrices using equations (6.26), (6.27) and
          (6.28).
        • Calculate the reprojection distance for each image point correspondence in
          all three views, using the formula (3.27). Compute the number of 3-view
          point correspondences with the maximal reprojection error below a preset
          threshold $t = 0.5$ pixel.
        • Choose the value of parameters $\delta_2$ and $\delta_1$ with the biggest support, that is
          with the largest number of inlying 3-view correspondences.
    (4) If i =1
        • Define camera matrices using equations (6.26), (6.27) and (6.28).
        • Calculate initial structure from all 3-view correspondences by linear
          triangulation.
       Else
        • Refine the last camera of the actual canonical camera triplet using the
          equation (6.32) and merge it with the existing camera matrices.
        • Fill the current structure with new 3D points acquired by triangulation using
          the last three cameras and 3-view correspondences of these views.
    (5) Increment i
```

Algorithm 6.2. Estimation of cameras' global rotations and translations to perform multiview structure reconstruction.

$$\mathbf{P}_{\mathbf{e\ i}} = [\mathbf{R}_\mathbf{i} \,|\, \mathbf{t}_\mathbf{i}] \tag{6.29}$$

as

$$\mathbf{P}_{\mathbf{e\ i+1}} = [\mathbf{R}_\mathbf{E}\, \mathbf{R}_\mathbf{i} \,|\, \mathbf{R}_\mathbf{E}\, \mathbf{t}_\mathbf{i} + \mathbf{t}_\mathbf{E}\,] \tag{6.30}$$

supposing the essential matrix between these views

$$\mathbf{E} = [\mathbf{t_E}]_\mathbf{x}\,\mathbf{R_E}\,. \tag{6.31}$$

Considering the knowledge of previously calculated parameters α_2 and δ_2(α_1 and δ_1 were set to 1), we can derive the formula specifying the fourth camera matrix

$$\mathbf{P_{e\,4}} = [\mathbf{R_{43}R_{31}} \mid \mathbf{R_{43}t_{31}} + \alpha_2\delta_2\,\mathbf{t_{n34}}], \tag{6.32}$$

which enables to place new camera within the existing reconstruction correctly.

A robust approach should be used again, as not all 3D points are suitable for the calculation of needed parameters. As a criterion for evaluation, reprojection error in 2D-space is applied. The summary of the global rotation and translation estimation is described in algorithm 6.2.

7 Tests on the synthetic data

In order to verify the functionality of the implemented algorithm, I generated simple synthetic 3D data formed by 18 space points creating a virtual object, see Fig. 7.1. and Fig. 7.2 Camera motion was simulated using 10 cameras with constant internal parameters.

The type of motion was selected so as to resemble the colonoscopic camera movement; the movement close to pure translation motion with small baseline between subsequent views. The rotations of cameras were generated randomly with limited angles between 0 and $\pi/20$. The translation camera motion was set to generate a movement with a constant distance between camera positions, the relative translations between subsequent cameras were set to have a constant norm value equal to one unit. The camera was calibrated to have the calibration matrix equal to identity matrix. An example of the camera trajectory can be seen in Fig. 7.3. Image projections of the space points were computed using the equation (3.6).

7.1 Experiments

Several experiments were performed. The 3D reconstruction of the virtual object was obtained from all triplets of the successive camera matrices. The quality of the reconstruction was validated through the mean 3D error - the mean distance between the true 3D points and the reconstructed 3D points over 10 trials.

At first, tests were performed repeatedly with exact values obtained from the projection equation (3.6) - without considering limited image resolution or noise. The correct setting of camera calibration parameters was used. The results show that the algorithm works correctly if the data are exact enough and the camera is truly calibrated, see Fig. 7.5.

In the second experiment, Gaussian noise was introduced both to the row and column coordinates of the 2D projected points. The noise amplitude was set to various levels, considering the fact that the images with the size 2x2 units have a resolution 512 x 512 pixels, see fig. 7.4.

First, experiments to test the robustness of the 5-point algorithm alone, not considering possible error contribution arising from inaccurate estimation of previous cameras, were conducted. Reconstruction was obtained using the first pair of cameras. The curve summarizing the mean 3D error of the robust five-point algorithm as a function of various Gaussian noise levels is shown in Fig. 7.6. The error curve increases basically linearly with the noise level. Some examples of these experiments are shown in Fig. 7.7 and 7.8.

As the next step, the mean 3D errors corresponding to partial reconstructions acquired by using Algorithm 6.2 were evaluated. The graph showing the mean 3D geometrical errors for the different noise amplitude is given in Fig. 7.9. Examples of some of these results are shown in Fig. 7.10, 7.11 and 7.12. It can be seen that the reconstructions acquired from cameras with higher sequence numbers are less accurate. These cameras were evaluated by means of previously computed camera matrices. The estimation error of the earlier cameras propagates to later cameras and increases the final reconstruction error.

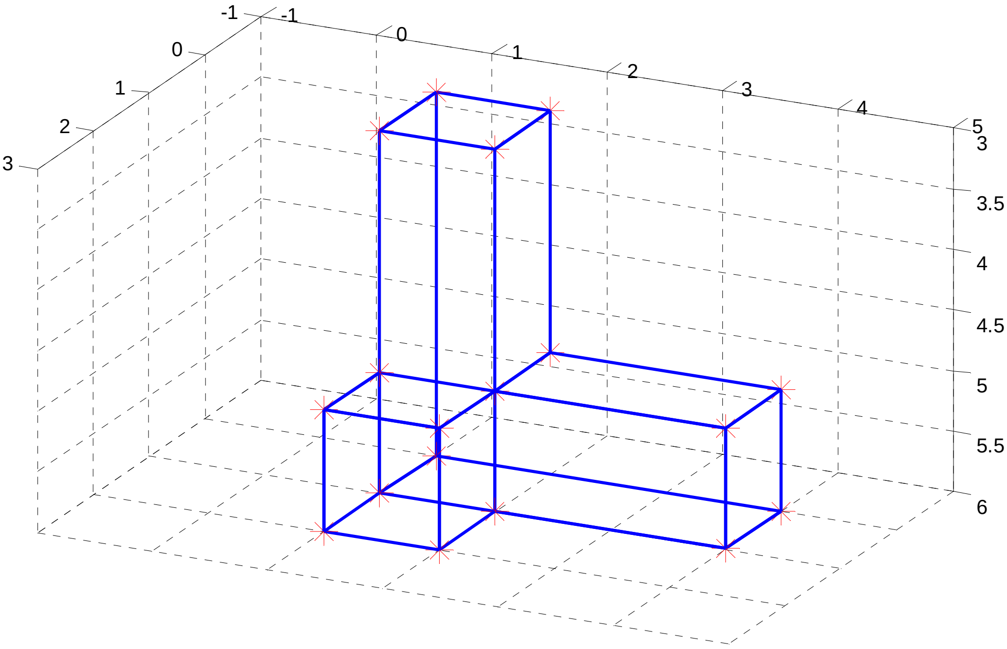


Fig. 7.1: Synthetic 3D object

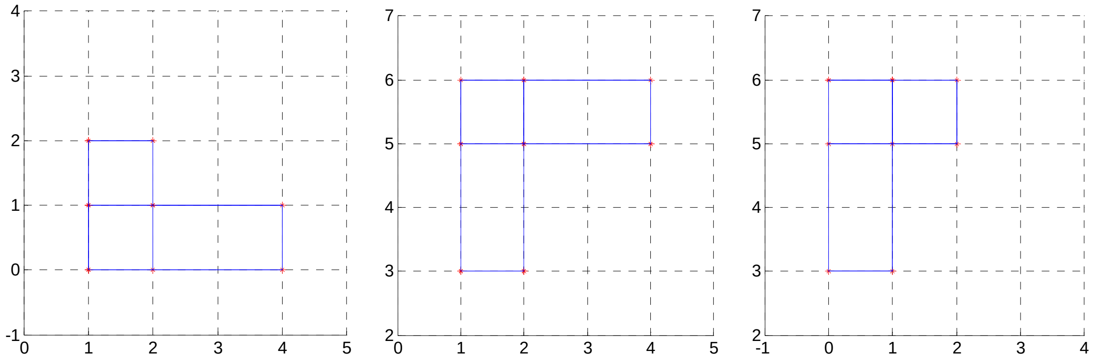


Fig. 7.2: Synthetic 3D data. Left image: X-Y view. Middle image: X-Z view. Right image: Y-Z view.

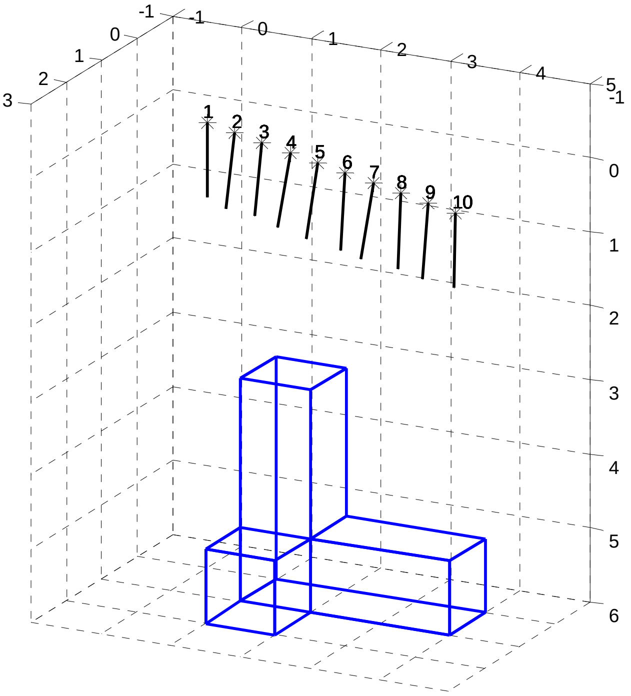


Fig.7.3: Camera movement along the 3D scene: black stars represent particular camera centers and black lines show directions of the camera optical axes.

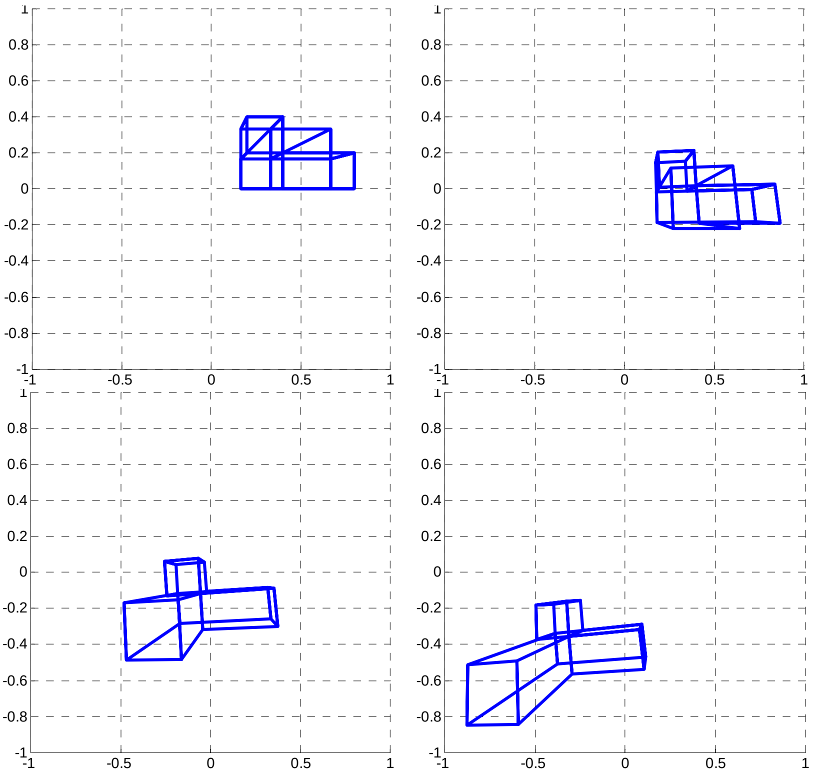


Fig. 7.4: The examples of the 3D scene projected to 2D. The images are supposed to have the resolution 512 x 512 pixels. Top images: views corresponding to the first and the second camera. Bottom images: views corresponding to the ninth and the tenth camera.

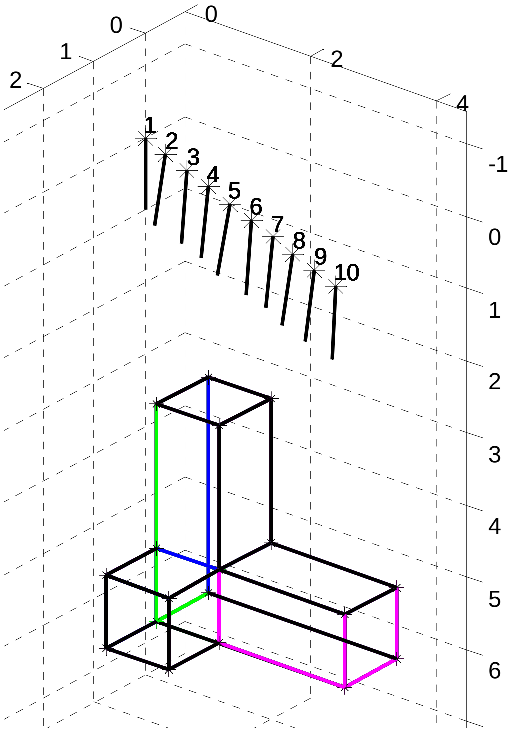


Fig. 7.5: A 3D reconstruction of the virtual object using correctly calibrated triplets of cameras: 1-2-3 (green), 4-5-6 (magenta) and 8-9-10 (black). No Gaussian noise was applied. The original 3D object is displayed in blue. The particular reconstructions overlap, thus the final scene is multicolored.

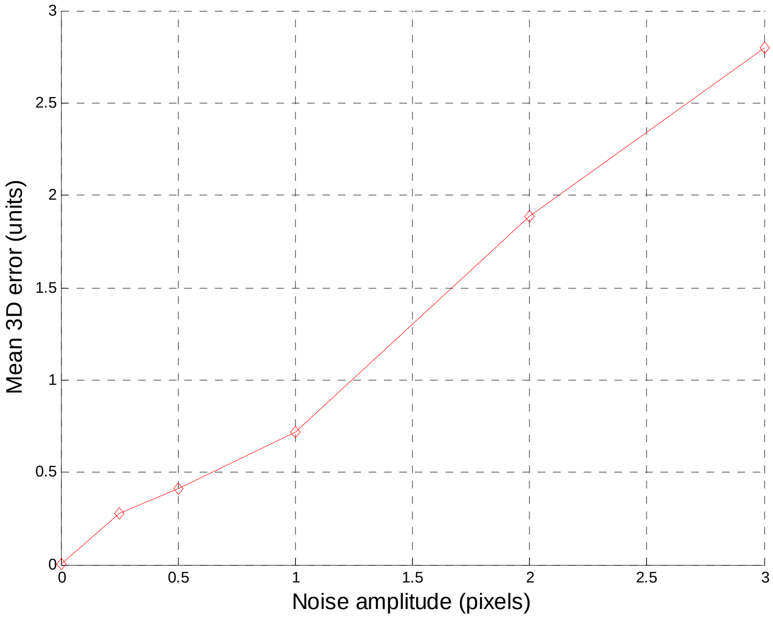


Fig. 7.6: The mean 3D error of the robust 5-point algorithm applied to a pair of images as a function of the Gaussian noise amplitude added to the 2D coordinates. The curve is evaluated over 10 trials.

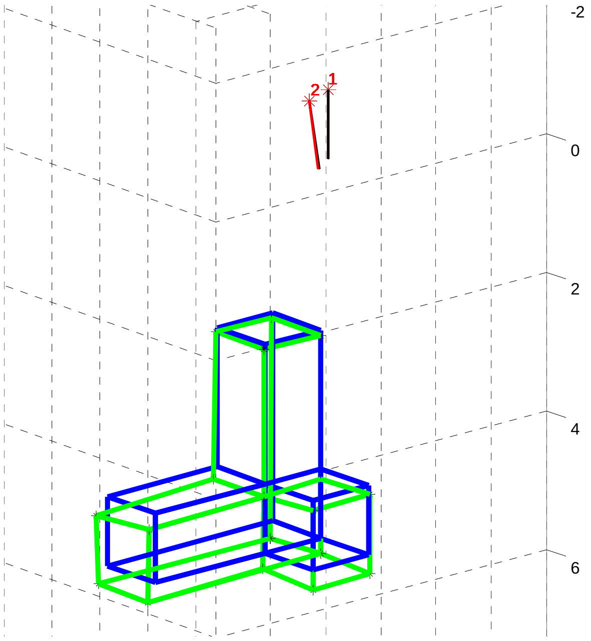


Fig. 7.7: A 3D reconstruction of the virtual object, the noise amplitude = 0.25 pixel. The first pair of cameras was used to compute the 3D scene in green. The original 3D object is displayed in blue. The figure illustrates the error of the robust 5-point algorithm when input data is noisy.

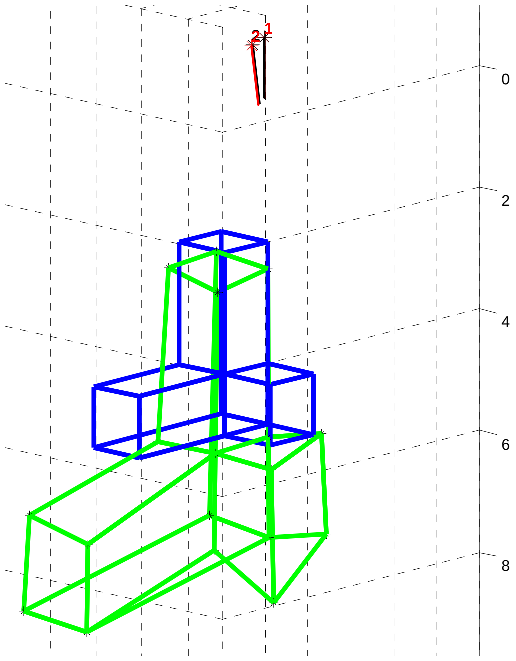


Fig. 7.8: The 3D reconstruction of the virtual object, the noise amplitude = 1 pixel. The first pair of cameras was used to compute the 3D scene in green. The original 3D object is displayed in blue. The figure illustrates the error of the robust 5-point algorithm when input data is noisy.

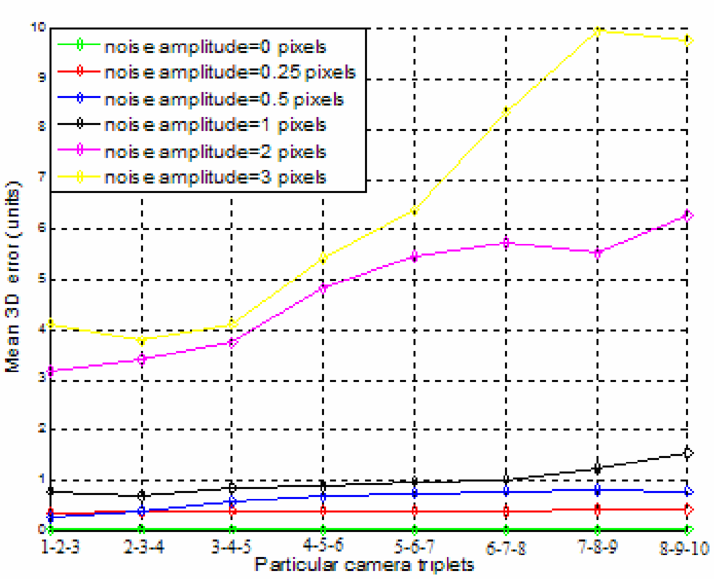


Fig. 7.9: A mean 3D error of reconstruction, which was acquired using triplets of the successive camera matrices. Different levels of Gaussian noise amplitude were added to the input 2D coordinates. Curves are evaluated over 10 trials.

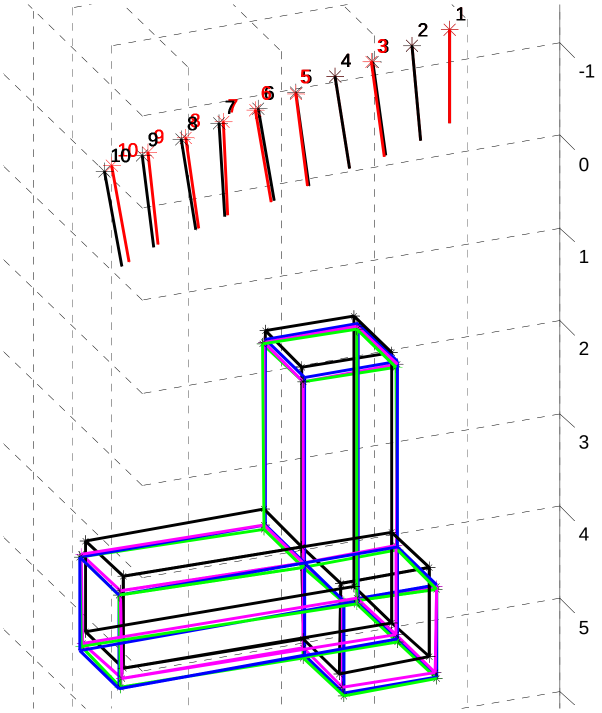


Fig. 7.10: A 3D reconstruction of a virtual object, the noise amplitude = 0.25 pixels. Calibrated triplets of cameras were used: 1-2-3 (green), 4-5-6 (magenta) and 8-9-10 (black). The original 3D object is displayed in blue. The real cameras are represented in black; the computed cameras are displayed in red.

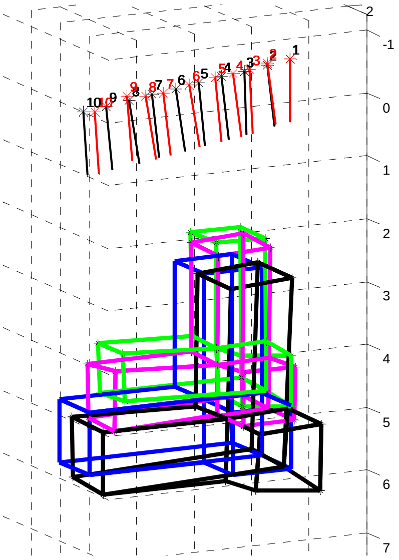


Fig. 7.11: A 3D reconstruction of a virtual object, the noise amplitude = 1 pixel. Calibrated triplets of cameras were used: 1-2-3 (green), 4-5-6 (magenta) and 8-9-10 (black). The original 3D object is displayed in blue. The real cameras are represented in black; the computed cameras are displayed in red.

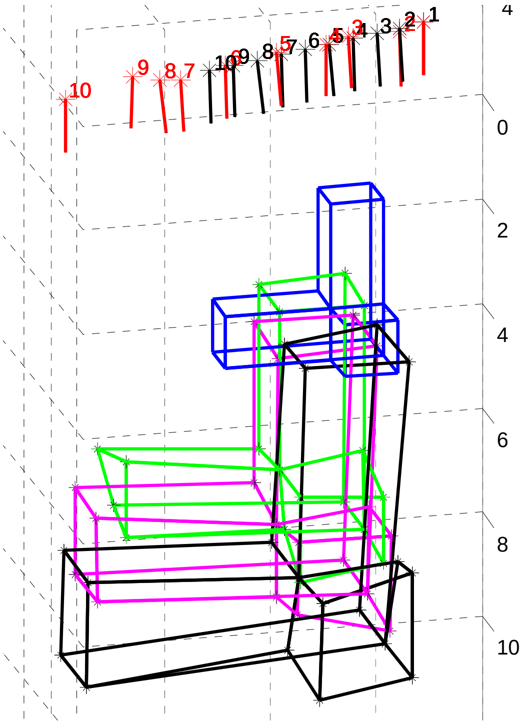


Fig. 7.12: A 3D reconstruction of a virtual object, the noise amplitude = 2 pixels. Calibrated triplets of cameras were used: 1-2-3 (green), 4-5-6 (magenta) and 8-9-10 (black). The original 3D object is displayed in blue. The real cameras are represented in black; the computed cameras are displayed in red.

7. 2 Results evaluation

The experiments show that the reconstruction error increases with the noise level. If the noise level exceeds 1 pixel, the reconstruction error rises significantly. Moreover, the error is amplified through the multiview reconstruction. The noise level is the decisive factor for the reconstruction error. The figures show, however, that the shape of the object is not deformed so much, but that the biggest error contribution comes from the fact that partial object models are reconstructed in different scale factors. The solution leading to a better reconstruction may consist in more accurate estimation of consistent global scale.

8 Reconstruction results using real colonoscopic data

In this chapter, results obtained by the suggested reconstruction algorithm applied to the input colonoscopic image sequence are presented.

Better reconstruction results have been achieved for a sequence consisting of more distant images and for image sequences consisting of a lower number of frames. Using all 30 frames from the input sequence, a large scale discrepancy of partial 3D colon models was present, as was found by experiments on synthetic data. Demonstration of the reconstruction error this is shown in Fig. 8.1. Structure is evaluated stepwise using camera triplets. The individual contributions of the structure reconstructed from successive camera triplets are depicted in different colors in the following order: green, red, blue, magenta and yellow. It can be seen that green and red point clouds corresponding to the first camera triplet are reconstructed with a larger scale than the other point clouds.

An example of a 3D model generated by the implemented algorithm is shown in Fig.8.2. The structure was reconstructed from 15 images selected from the input image sequence consisting of 30 images such that every other frame was taken. Focal-length was set to value 250. I have tried several focal-length values ranging from 200 to 300, however no significant difference in quality of reconstruction was observed. The low number of correspondences (mainly due to a larger distance between successive frames) used for estimation of the space points and a lower number of frames result in a poor structure (this is more obvious from corresponding Vrml model presented on enclose CD). A relatively large reconstruction inaccuracy is evident.

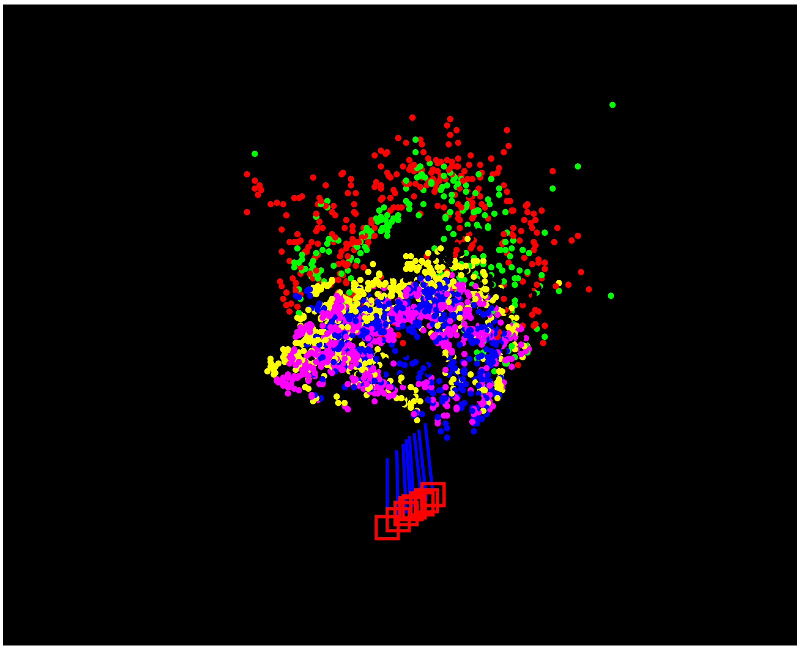
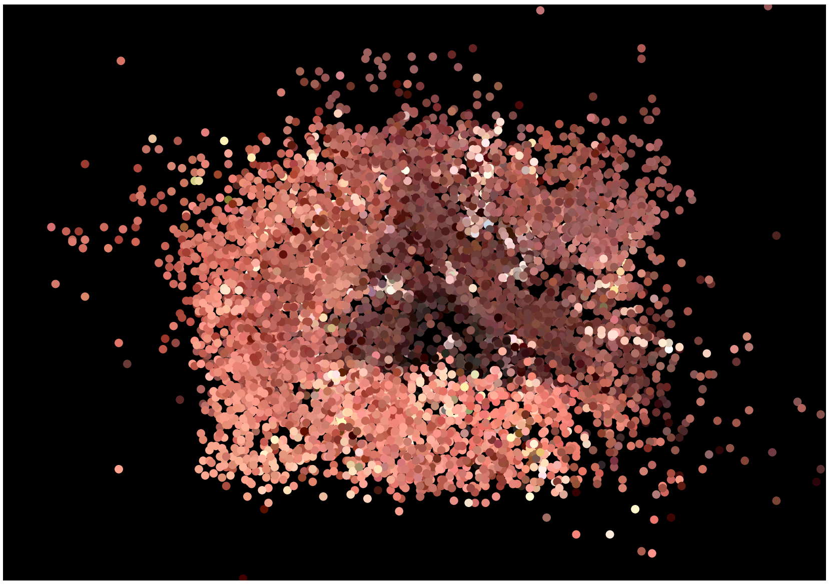
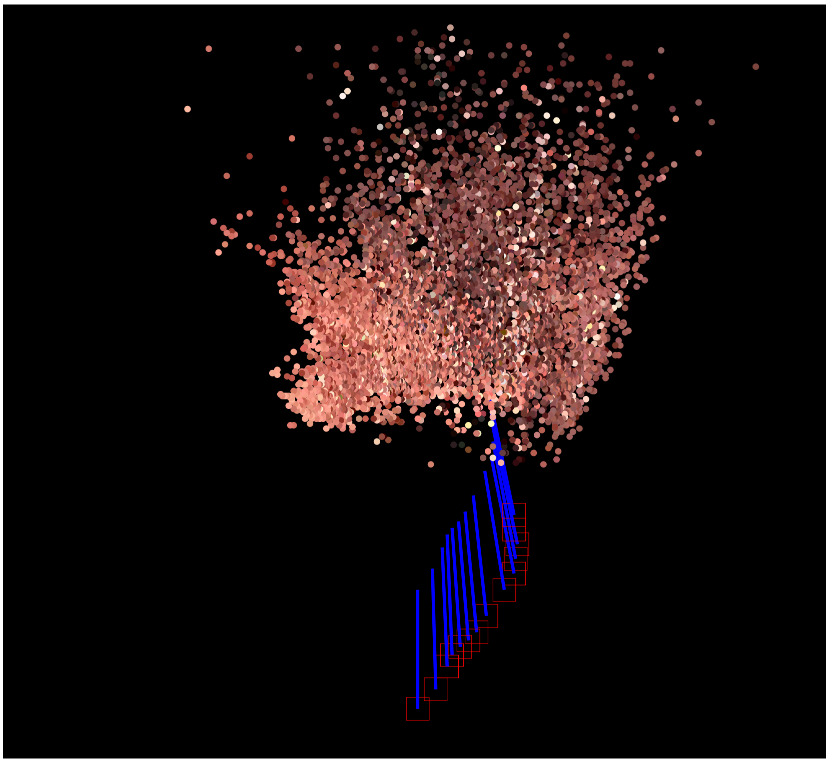


Fig. 8.1. Merging the colon structure.



(a)



(b)

Fig. 8.2. Reconstruction of the colon using 15 views obtained from the input colonoscopic sequence such that every other frame was taken. Focal-length was set to 250. (a) Front view; (b) Side view. Red squares represent camera centers, blue lines show optical axes of cameras.

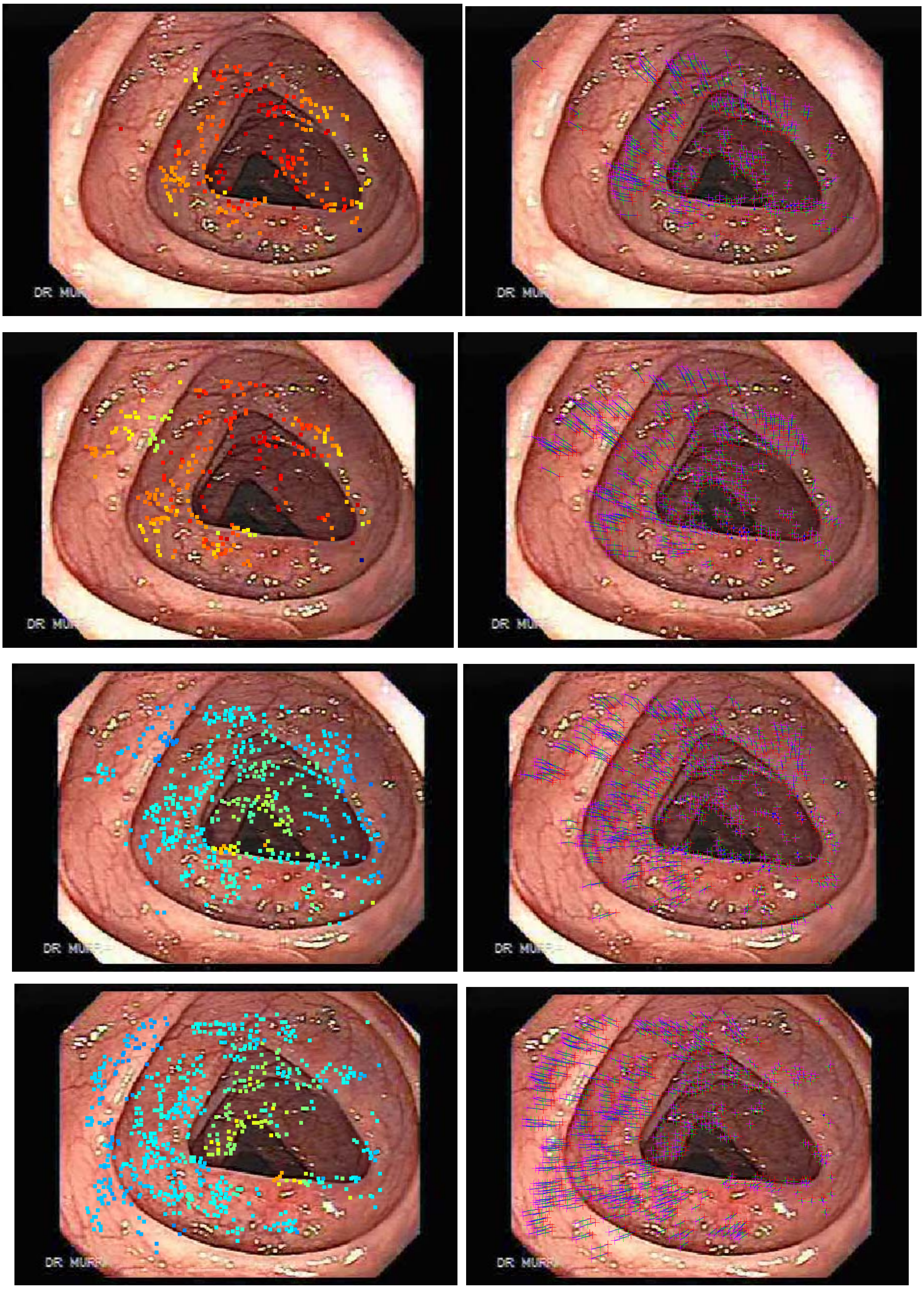


Fig 8.3. Illustration of depths of 3D points (left column) corresponding to tracks in the image frames (right column).

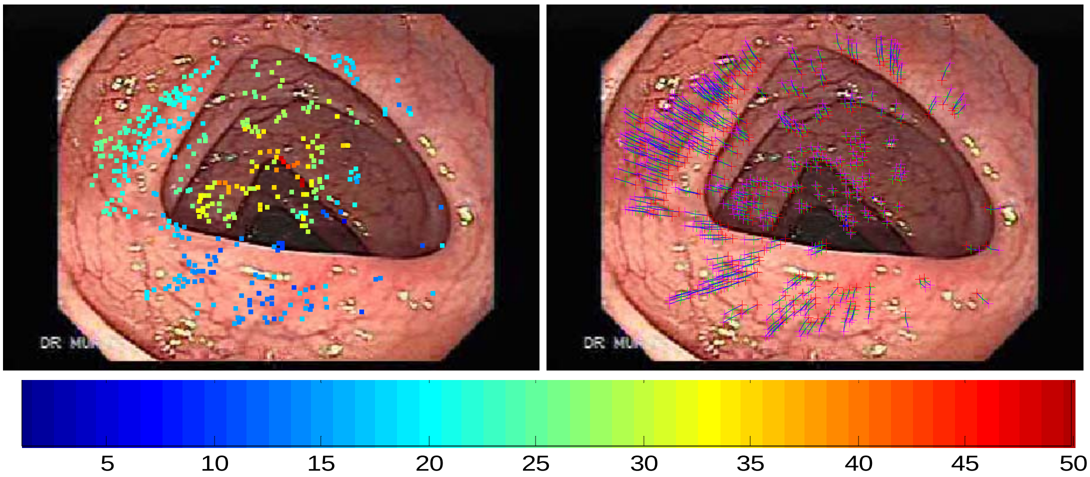


Fig 8.3. Illustration of depths of 3D points (left column) corresponding to tracks in the image frames (right column). Depth is represented in color spectrum from blue to red – see the color bar above. The nearest points are depicted in dark blue, the furthermost in dark red. The image frames are ordered (in terms of camera forward motion) from top to bottom.

Accuracy of the reconstruction may be well visually qualified using colored representation of the depths of space points. Depiction of depths of points calculated from three-view feature correspondences through the reconstruction process shown in Fig. 8.2 is given in Fig. 8.3. Distinctive scale discrepancy can be identified in the second and the third view when depths of tracked points change inadequately with respect to camera displacement. Colors of points in the third view show that points are considerably closer to camera in comparison to points in the second view. That is supposedly caused mainly by the scale discrepancy between partial structures reconstructed from image points in the second and the third view. As a result, all points shown in the third view are closer to camera. Still, it can be seen from Fig. 8.4 that estimation accuracy of relative depths of points (considering only partial structures) is basically the same throughout the reconstruction process, from aside the scale discrepancy (larger depths are generally assigned to more distant points).

Considering this results, I have tried to prevent a large reconstruction error by using a manual approach to set a scale factor of relative translations between successive views. The setting was done by estimating a camera displacement between successive frames and by visual examination of depths of reconstructed points. This approach is only approximate in principle, yet large scale discrepancy can be eliminated in this way.

An example of such generated colon model is shown in Fig. 8.4. Visualization of depths of points evaluated through this reconstruction process is shown in Fig. 8.5. The mean reprojection error calculated for a fully automatic reconstruction given in Fig. 8.2 and the reconstruction utilizing a manual setting of scale factors of relative translations shown in Fig. 8.4 was calculated, showing that the mean reprojection error decreased from value 0.4938 px to 0.1345 px after applying a manual approach. Note that small reprojection errors in this case do not guaranty small reconstruction error in 3D.

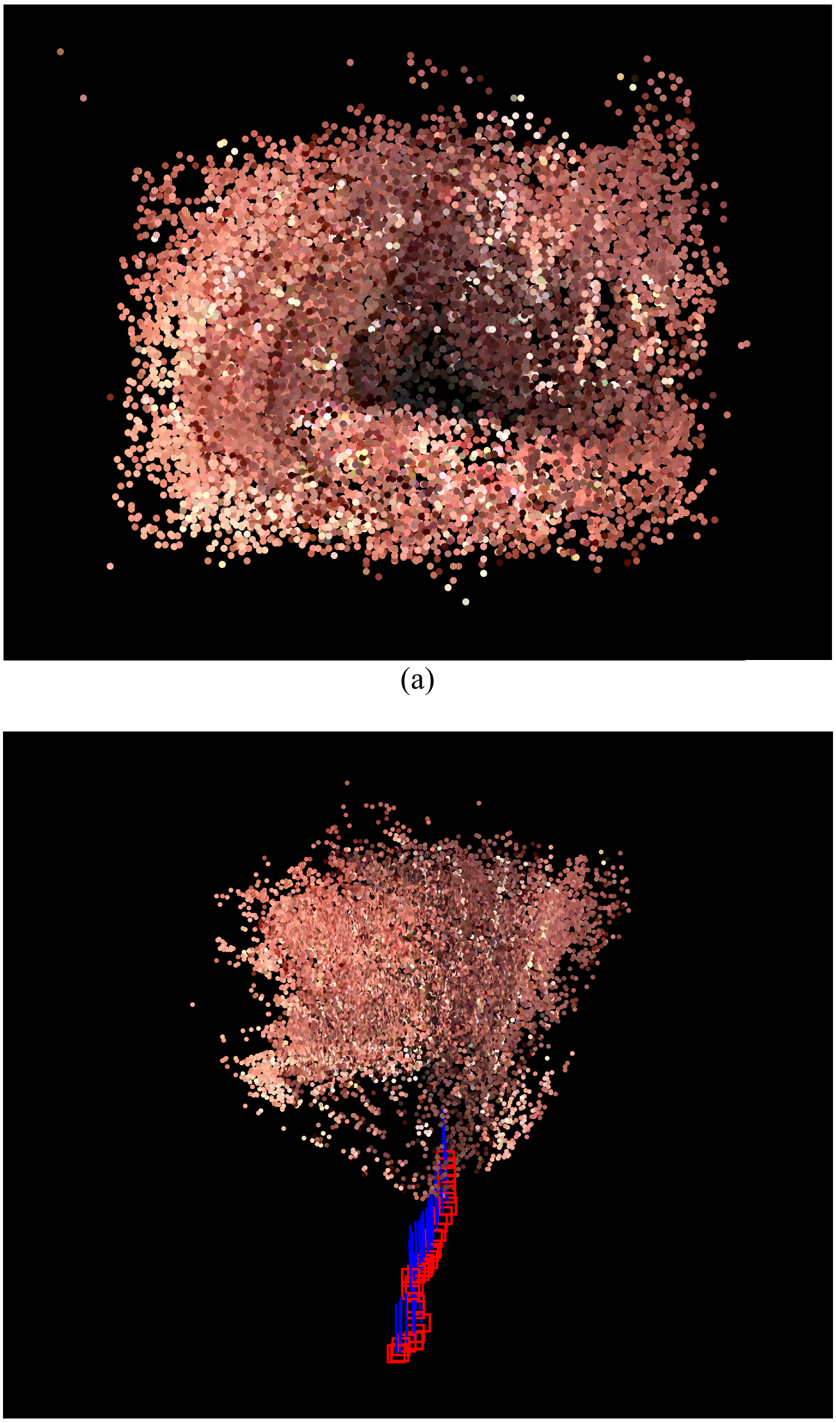


Fig. 8.4. Reconstruction of the colon using manual settings of scale factors of relative translations between successive views. Input image sequence consists of 30 frames. Focal-length was set to value 250. (a) Front view; (b) Side view. Red squares represent camera centres, blue lines show optical axes of cameras.

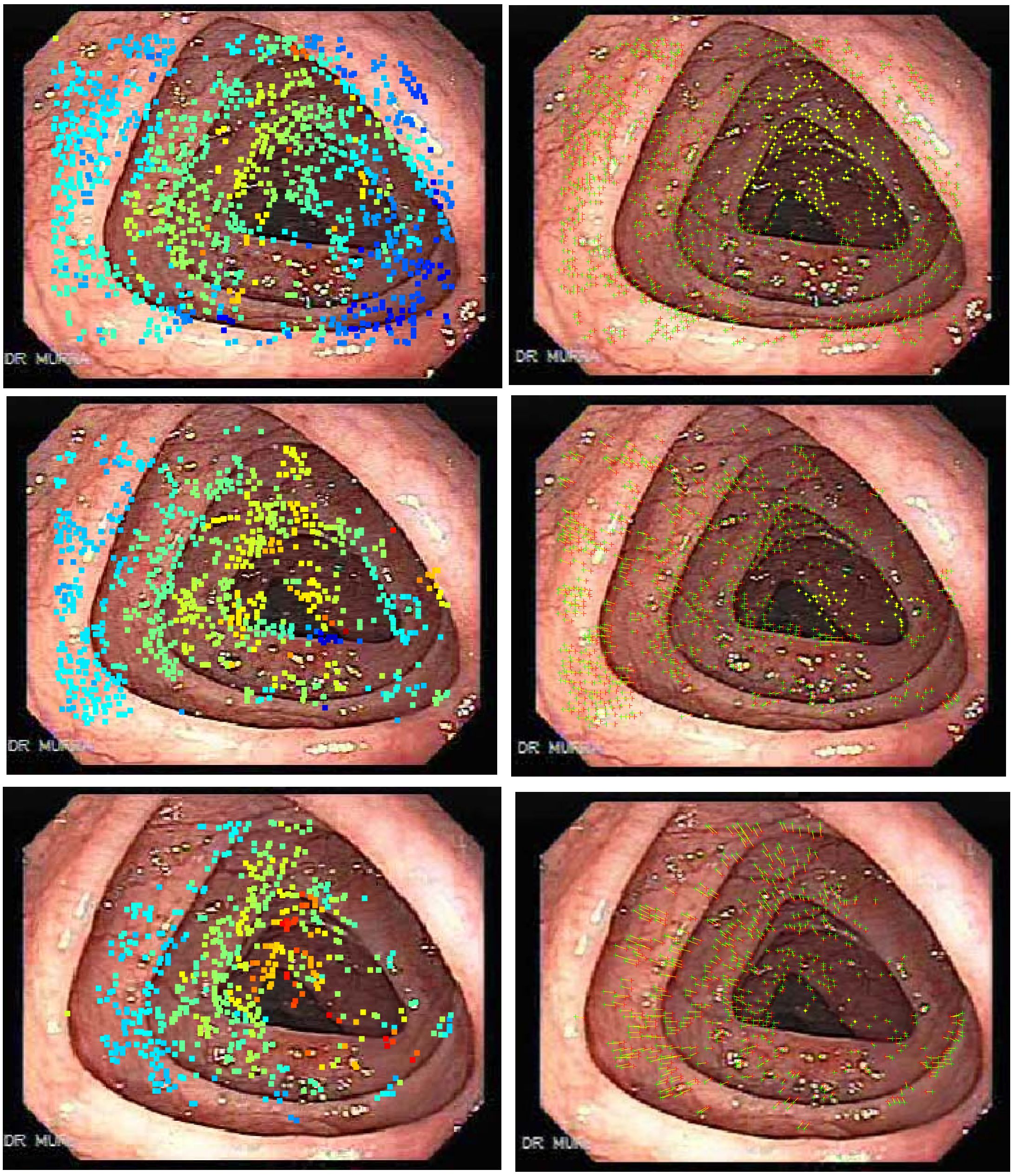


Fig 8.5. Illustration of depths of 3D points (left column) corresponding to tracks in the image frames (right column).

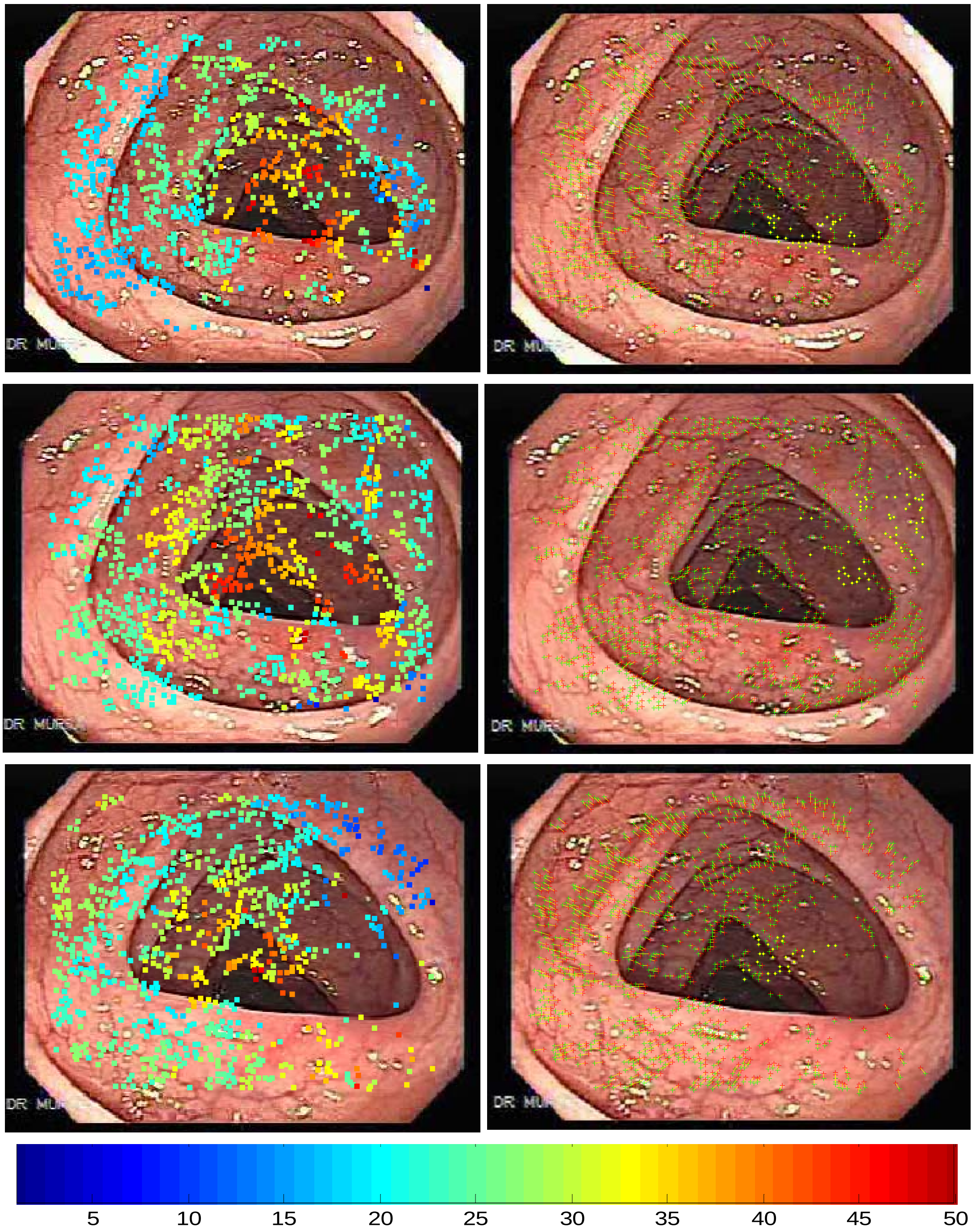


Fig 8.5. Illustration of depths of 3D points (left column) corresponding to tracks in the image frames (right column). Depth is represented in color spectrum from blue to red – see the color bar above. The nearest points are depicted in dark blue, the furthermost in dark red. The image frames are ordered (in terms of camera forward motion) from top to bottom.

9 Conclusion

The aim of this work have been to project and implement an algorithm for automatic sparse 3D reconstruction from 2D image sequence obtained during a colonoscopy. A 3D model of the colon is a needed application in surgical training and diagnosis specification. This model can be of course beneficial only on the condition that the reconstruction is detailed and of sufficient quality.

In my background research on implemented systems focusing on the automatic 3D reconstruction from colonoscopic videosequence, I have found merely one system dealing with this problem, published in article [8]. Unlike the algorithm in my thesis, this one focuses on modeling local anatomical structures (for instance a polypus or a tumor), not more extensive part of the colon surface. Moreover, the system operates with known configuration of camera internal parameters, whereas for our algorithm, the information about the focal length parameter needs to be evaluated using auto-calibration process.

To my knowledge, the approach of 3D reconstruction from colonoscopic images is not currently in common use in medicine, the reasons probably being that creating a system which provides a fully automatic 3D colon reconstruction of sufficient reliability and efficiency seems to be a very demanding task. Using theoretical knowledge from literature, I attempted to implement a system performing a sparse 3D colon model from a short colonoscopic sequence.

The functionality of the system was tested on a simple synthetic object. I have confirmed that the algorithm operates correctly (with error equal to zero) on the condition that the input data do not contain any noise and the camera is accurately calibrated. In case the input data are noisy, the error of the algorithm increases. For noise level exceeding 1 pixel, there is already a considerable error. The most responsible for the reconstruction error is probably the scale discrepancy of partial 3D models, which are successively glued in the final 3D model. As the later cameras are evaluated by means of previously computed cameras, the error of the earlier cameras is reflected in the estimation of the following cameras, and the error propagates into the entire program chain.

Results on real colonoscopic data confirmed the findings of experiments on synthetic data that a large scale discrepancy of partial 3D models is often present. However, using shorter sequences and sequences consisting of more distant images, the obtained reconstruction results have been more accurate and relatively stable. I have tried to resolve a part of this problem by rejecting the idea of a fully automatic reconstruction and introducing an implementation utilizing manual setting of the norm of relative translations between successive cameras instead. The setting was performed estimating a camera displacement in the input image sequence and with reference to visual examination of reconstructed model parts. As a result, this solution is also approximate in principle, yet large errors can be eliminated in this way. The improvement is obvious if we compare the final reconstruction to the previous, fully automatic one.

Several factors may contribute to the fact that the application work with a considerable error. Firstly, it is probably a relatively low quality of the input images arising from the resolution of 240x352 pixels and the fact that some frames are slightly blurred. Secondly, an absence of knowledge about the camera focal length parameter, when only the approximate value was used, probably increases the final error. Moreover, it is necessary to take into account the fact that the applied algorithm, when compared to the real-life sophisticated systems implemented by a team of company developers or research groups, is a relatively simple one so that it can be accomplished within a thesis. The complex methods necessary for high-quality 3D reconstruction require a number of optimalization techniques and corrections if undesirable errors are to be eliminated.

Despite the fact that the quality of 3D reconstruction acquired by the algorithm applied in this thesis can hardly be compared to needs in practice, I believe I can state that it creates a solid and usable framework for further development and improvement.

9.1 Suggestions for future work

There are many ways in which the algorithm can be improved. First of all, my recommendation is to use the camera with higher resolution and internal parameters known in advance. A more sophisticated system for feature detection and tracking could be used to achieve improvement, for instance affine-invariant MSER detector. Feature tracking should be performed over as many frames as possible, in order to improve accuracy, and nonlinear optimalizations should be applied. Also, using a more effective algorithm for evaluating global translations, for instance based on the solution using L_∞-norm, should be of considerable help. And of course, a crucial component of the application which still needs implementing is the so far absent dense reconstruction part.

Bibliography:

[1] R.I.Hartley, A. Zisserman. Multiple View Geometry in Computer Vision, Cambridge U. Press, 2000.

[2] M. Pollefeys - L. Van Gool - M. Vergauwen - F. Verbiest - K. Cornelis - J. Tops - R. Koch. Visual modeling with a hand-held camera, International Journal of Computer Vision 59(3), 207-232, 2004.

[3] Marc Pollefeys, Maarten Vergauwen and Luc Van Gool. "Automatic 3D Modeling from Image Sequences," In ISPRS, vol. 33, Amsterdam, 2000.

[4] Tomas Rodriguez, Peter Sturm, Marta Wilczkowiak, Adrien Bartoli, Matthieu Personnaz, Nicolas Guilbert, Fredrik Kahl, M. Johansson, Anders Heyden, José Manuel Menendez, José Ignacio Ronda, Fernando Jaureguizar. "Visire. Photorealistic 3D reconstruction from video sequences". In IEEE International Conference on Image Processing, Barcelona, Spain, vol. 33, pp. 705-708, September 2003.

[5] G. Larchev, E. Ng, N. Williams. "Extraction of Topological Data from Aerial Images", EE368 Digital Image Processing Final Project, June 2001.

[6] A. E. Kaufman. 3D virtual colonoscopy project.
http://www.cs.sunysb.edu/~vislab/volvis_home.html

[7] S. Seitz. Lecture notes on 3d photography, 2000.
http://www.cs.cmu.edu/~seitz/course/Sigg00/notes.html

[8] Koppel Dan, Chen Chao-I., Wang Yuan-Fang, Lee Hua, Gu Jia, Poirson Allen, Wolters Rolf. Toward automated model building from video in computer-assisted diagnoses in colonoscopy. Proceedings of the SPIE, Volume 6509, pp. 65091L, 2007.

[9] Wei Li, Arie Kaufman, and Kevin Kreeger. Real-Time Volume Rendering for Virtual Colonoscopy. Proceedings of Volume Graphics, p. 363, 2001.

[10] A.W. Fitzgibbon, G. Cross, and A. Zisserman. Automatic 3D model construction for turntable sequences. In Proc 3D Structure from Multiple Images of Large-Scale Environments, pages 155–170, 1998.

[11] T. Werner. Korespondence sekvence obrazů planární scény. Computer vison and virtual reality course. http://cmp.felk.cvut.cz/cmp/courses/PVR/2007/Labs/DU-02.html

[12] Daniel Martinec, Tomas Pajdla. Robust rotation and translation estimation in multiview reconstruction. In Proceedings of the Computer Vision and Pattern Recognition conference 2007, IEEE, Minneapolis, MN, USA, June 2007.

[13] C. Harris and M.J. Stephens. A combined corner and edge detector. In Alvey Vision Conference, pages 147–152, 1988.

[14] T. Werner. Harris corner detector. Computer vision and virtual reality - course material. 2007.
http://cmp.felk.cvut.cz/cmp/courses/PVR/2007/Labs/harris.pdf

[15] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F Schaffalitzky, T. Kadir, and L. van Gool. A comparison of affine region detectors. International Journal of Computer Vision, 65(7):43 - 72, November 2005.

[16] Brendan McCane. Research in feature tracking. 2000.
http://www.cs.otago.ac.nz/research/vision/Research/FeatureTracking/featuretracking.html

[17] T. Pajdla. Computer vision for informatics. Course material, 2006.
http://cmp.felk.cvut.cz/cmp/courses/pvi2006/Lecture/PVI-2006-Lecture-05-06.pdf

[18] O. Faugeras, Q.-T. Luong, and S. J. Maybank. Camera Self-Calibration: Theory and Experiments. In ECCV, pages 321–334, Santa Margherita Ligure, Italy, May 1992.

[19] Q.-T. Luong and O. Faugeras. Self-calibration of a moving camera from point correspondences and fundamental matrices. IJCV, 22(3):261–289, Mar. 1997.

[20] M. Pollefeys and L. Van Gool. Stratified Self-Calibration with the Modulus Constraint. PAMI, 21(8):707–724, Jan. 1999.

[21] R. Hartley, E. Hayman, L. de Agapito, and I. Reid. Camera calibration and the search for infinity. In ICCV, volume 1, pages 510–517, Kerkyra, Greece, Sept. 1999.

[21] A. Heyden and K. Aström. Euclidean Reconstruction from Constant Intrinsic Parameters. In ICPR, volume 1, pages 339–343, Vienna, Austria, Aug. 1996.

[22] B. Triggs. Autocalibration and the absolute quadric. In CVPR, pages 609–614, San Juan, Puerto Rico, June 1997.

[23] F. Kahl, B. Triggs, K. Astrom, Critical motions for auto-calibration when some intrinsic parameters can vary, J. Math. Imaging Vision 13 (2) (2000) 131–146.

[24] M. Pollefeys, R. Koch, L. Van Gool, Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters, in: Proc. 6th Internat. Conf. on Computer Vision, Bombay, India, 1998, pp. 90–96.

[25] P. Sturm, Critical motion sequences for monocular self-calibration and uncalibrated Euclidean reconstruction, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Puerto Rico, June, 1997, pp. 1100-1105.

[26] M. Pollefeys, Self-calibration and metric 3D reconstruction from uncalibrated image sequences, PhD. thesis, K.U.Leuven, 1999.

[27] P. Sturm, On Focal Length Calibration from Two Views, IEEE International Conference on Computer Vision and Pattern Recognition, Volume 2, pp. 145–150, 2001.

[28] P.Sturm,et al, Focal length calibration from two views: method and alaysis of singular cases, Computer Vision and Image Understanding,Vol 99, No.1, 2005.

[29] H.Stewénius, D.Nistér, F.Kahl, F.Schaffalitzky, A minimal solution for relative pose with unknown focal length, in Proc. IEEE-CVPR-2005, 2005.

[30] Hongdong Li, A Simple Solution to the Six-point Two-view Focal-length Problem. In Proc. ECCV 2006.

[31] S.Petitjean, Algebraic geometry and computer vision: Polynomial systems, real and complex roots, Journal of Mathematical Imaging and Vision,10:191-220,1999.

[32] D. Cox, J. Little and D. O'Shea, Ideals, Varieties, and Algorithms, ISBN 0-387-94680-2, Springer-Verlag, 1997.

[33] D.Cox, J.Little and D.O'shea, Using Algebraic Geometry, 2nd Edition, Springer, 2005.

[34] E. Kruppa, Zur Ermittlung eines Objektes aus zwei Perspektiven mit Innerer Orientierung, Sitz.-Ber. Akad. Wiss., Wien, Math. Naturw. Kl., Abt. IIa., 122:1939-1948, 1913.

[35] O. Faugeras, Three-Dimensional Computer Vision: a Geometric Viewpoint, MIT Press, ISBN 0-262-06158-9, 1993.

[36] S. Maybank, Theory of Reconstruction from Image Motion, Springer-Verlag, ISBN 3-540-55537-4, 1993.

[37] A. Heyden and G. Sparr, Reconstruction from Calibrated Cameras - a New Proof of the Kruppa-Demazure Theorem, Journal of Mathematical Imaging & Vision, 10:1-20, 1999.

[38] J. Philip, A Non-Iterative Algorithm for Determining all Essential Matrices Corresponding to Five Point Pairs, Photogrammetric Record, 15(88):589-599, October 1996.

[39] Nistér, D., 2004. An Efficient Solution to the Five-Point Relative Pose Problem, IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(6):756-770.

[40] H. Stewénius, Christopher Engels, D. Nistér. Recent Developments on Direct Relative Orientation, In ISPRS, vol. 60, pp 284-294, 2006.

[41] H. Stewénius. Matlab code for solving the calibrated five-point solver. http://www.maths.lth.se/~stewe/FIVEPOINT/.

[42] M. Bujňák. Dense Reconstruction from uncalibrated video. Rigorous Thesis. Comenius University, Bratislava, 2005.

[43] H. Cornelius, R.Šára, D. Martinec, T. Pajdla, O. Chum, and J. Matas. Towards complete free-form reconstruction of complex 3d scenes from an unordered set of uncalibrated images. In Statistical Methods in Video Processing, pages 1-12, Berlín, Německo, 2004. Springer.

[44] F. Kahl. Multiple view geometry and the L_∞-norm. In ICCV05, pp. II: 1002–1009, 2005.

[45] L. Kunc. Automatická 3D rekonstrukce z 2D snímku otácejícího se objektu. Master thesis. Czech Technical University in Prague, 2007.

[46] Marc Levoy, Jeremy Ginsberg, Jonathan Shade, Duane Fulk, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, "The Digital Michelangelo Project: 3D Scanning of Large Statues," Proceedings of the 27th annual conference on Computer graphics and interactive techniques, 2000, pp.131-144.

[47] Gabriele Guidi, Laura Micoli, Michele Russo, Bernard Frischer, Monica De Simone, Alessandro Spinetti, Luca Carosso, "3D digitization of a large model of imperial Rome," Fifth International Conference on 3-D Digital Imaging and Modeling, 2005, pp.565-572

[48] J. Matas, O. Chum, M. Urban, T. Pajdla: Robust wide baseline stereo from maximally stable extremal regions. In: BMVC. (2002) 384–393.

[49] Chum, O., Matas, J., Kittler, J.: Locally optimized RANSAC. In: DAGM. (2003) 236–243.

[50] Šára, R. Finding the largest unambiguous component of stereo matching. In: ECCV. (2002) 900–914

[51] Daniel Martinec, Tomás Pajdla: 3D Reconstruction by Gluing Pair-Wise Euclidean Reconstructions, or "How to Achieve a Good Reconstruction from Bad Images". 3DPVT 2006: 25-32

[52] A. W. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed or open image sequences. In Proc. ECCV, pages 311{326, 1998.

[53] R. I. Hartley. Euclidean reconstruction from uncalibrated views. In J. Mundy, A. Zisserman, and D. Forsyth, editors, Applications of Invariance in Computer Vision, LNCS 825, pages 237-256. Springer-Verlag, 1994.

[54] W. Lorensen and H. Cline. Marching cubes: A high resolution 3d surface construction algorithm. ACM Computer Graphocs, 21(24):163-169, July 1987.

[55] J. Matas, Š. Obdržálek, and O. Chum. Local affine frames for wide-baseline stereo. In ICPR(4), pp. 363–366, 2002.

[56] O. Chum, T. Werner, and J. Matas. Two-view geometry estimation unaffected by a dominant plane. In CVPR, vol. 1, pp. 772–779, 2005.

[57] D. Martinec and T. Pajdla. 3d reconstruction by fitting lowrank matrices with missing data. In Proc CVPR, vol. I, pp. 198–205, San Diego, CA, USA, June 2005.

[58] D. Koppel, Y. F. Wang, and H. Lee, Image-Based Rendering and Modeling in Video-Endoscopy," in Proc. of IEEE Int. Symp. on Biomedical Imaging, pp. 272-279, April 2004.

[59] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. Int. J. Computer Vision 60, pp. 91-100, 2004.

[60] G. Farin. Curves and Surfaces for Computer Aided Geometric Design, Academic Press, San Diego, CA, 1988.

[61] P. Besl, N. D. McKay. A Method for Registration of 3-D Shapes. IEEE Trans. Pattern Analy. Machine Intell. 14, pp. 239-256, 1992.

[62] A. Gyaourova. C. Kamath. SC. Cheung. Block Matching for Object Tracking. UCRL-TR-200271. 2003.

[63] Koch, R., Pollefeys, M., and Van Gool, L., 1998. Multi Viewpoint Stereo from Uncalibrated Video Sequences. Computer Vision - ECCV'98, LNCS, 1406, Springer-Verlag, pp.55-71.

[64] Jiří Žára, Bedřich Beneš, Jiří Sochor, Petr Felkel. Moderní počítačová grafika (2. vydání). Computer Press, 2005.

[65] K.Mikolajcyk, C. Schmid. An affine invariant interest point detector. In Proceedings of the 8th International Conference on Computer Vision, pp.128-142, 2002.

A The contents of enclosed CD

Results from my work are included on enclosed compact disc.

You can find there:

- Original colonoscopic videosequence.
- Colonoscopic sequence used as the input for this work
- The text of this master thesis in pdf format.
- Matlab files for visualization of reconstructed scene.
- Vrml files of reconstructed scene.
- Vrml animations of reconstructed scene.
- Matlab source code of the implemented system.

B Algorithm implementation

In this chapter, implementation and application of the algorithm suggested in this work is described. The algorithm is implemented in Matlab version 7.0.1. The final 3D models are also represented in VRML format.

Final results using the real data discribed in Chapter 8 are presented in the following files:

- `Model_man.m` – 3D matlab model presented in Fig. 8.4.
- `Depths_of_points_man.m` shows depths of 3D points represented in Fig. 8.5.
- `Colon_man_anim.wrl` – animation of vrml model presented in Fig. 8.4.
- `Colon_man.wrl` - vrml model presented in Fig. 8.4.
- `Model_aut.m` – 3D matlab model presented in Fig. 8.2.
- `Depths_of_points_aut.m` shows depths of 3D points represented in Fig. 8.3.
- `Colon_aut_anim.wrl` - animation of vrml model presented in Fig. 8.2.
- `Colon_aut.wrl` - vrml model presented in Fig. 8.2.

The main modules of the suggested algorithm are as follows.

- `[corresp,p_matches]=get_2view_tracks_tm(images)` performs template matching technique and epipolar geometry (EG) estimation for successive image pairs in the input image sequence. Epipolar geometry is estimated using a robust 8-point algorithm.

  `images` - input image sequence

  `corresp` - 2-veiw correspondences for successive image pairs estimated by EG

  `p_matches` - 2-veiw putative matches for successive image pairs estimated by template matching technique

  Example usage:

  ```
  clear all; close all;
  % retrieve variable images containing an input image sequence
  % consisting of 4 views
  images=load('im_sequence.mat');
  % retrive and show correspondences and putative matches and for
  % successive image pairs
  [corresp,p_matches]=get_2view_tracks_tm(images);
  ```

- `[corresp,p_matches]=get_2view_tracks_cbt(images)`performs correspondence based matching and epipolar geometry (EG) estimation for successive image pairs in the input image sequence. Epipolar geometry is estimated using a robust 8-point algorithm.

  `images` - input image sequence

  `corresp` - 2-veiw correspondences for successive image pairs estimated by EG

  `p_matches` - 2-veiw putative matches for successive image pairs estimated by correspondence based matching

```
Example usage:

clear all; close all;
% retrieve variable images containing an input image sequence
% consisting of 4 views
load('im_sequence.mat');
% retrive and show correspondences and putative matches for
% successive image pairs
[corresp, put_matches]= get_2view_tracks_cbt(images);
```

- `[Str_points,Colour]=MultiviewReconstruction(images,tracks3)` performs automatic multiview sparse reconstruction.

  `images` - input image sequence

  `tracks3` - 3-veiw correspondences for successive image triplets

  `f` – focal length value

```
Example usage:

clear all; close all;
% retrieve variable images containing an input image sequence
load ColonSeq
% retrive 3-view correspondences saved in variable tracks3
load ThreeViewTracks
% set focal length value
f=250;
[Str_points,Colour]=MultiviewReconstruction(images,tracks3,f);
```

- `[Str_points,Colour]=MultiviewReconstr_PartModelsScaled(multParameters)` performs multiview sparse reconstruction with hand-adjusted scale factors of relative translations between succesive views

  `multPar` – structure containing information about pair-wise metric reconstructions (relative rotations, relative translations, relative cameras, 2-view correspondences, images and calibration matrix).

  `Str_points – reconstructed 3D points`

  `Colour – information about 3D points’ color`

```
Example usage:

clear all; close all;
% retrieve pair-wise reconstruction parameters
load MultParameters
multPar = multiviewParameters;
[Str_points,Colour]=MultiviewReconstr_PartModelsScaled(multPar);
```

- `[Parameters]=MultiviewRelativeOrient(images,tracks,f)` retrives relative orientations between successive images in sequence and saves the following parameters to struct: relative oriantations, relative translation, 2-view correspondences, images, epipoles and calibration matrix.

  `images` - input image sequence

  `tracks` - 2-veiw correspondences for successive image pairs

  `f` – focal length value

`Parameters -` output variable is used as an input to function `MultiviewReconstr_PartModelsScaled`

```
Example usage:

clear all; close all;
% retrieve variable images containing an input image sequence
load ColonSeq
% retrive 2-view correspondences saved in variable tracks
load TwoViewTracks
% set focal length value
f=250;
[Parameters]=MultiviewRelativeOrient(images,tracks,f)
```

- `[foc]=FocalLengthRoots(FocImPair,FocTracks)`estimates possible 15 possible solutions for focal-length from randomly selected 6 two-view tracks
  `FocImPair -` input image pair
  `FocTracks –` 2-view correspondences detected on the image pair
  `foc -` 15 possible solutions for focal-length

```
Example usage:

clear all; close all;
% retrieve image pair
load FocTracks
% retrieve 2-view correspondences
load FocImPair
[foc]=FocalLengthRoots(FocImPair,FocTracks);
```

- `FocalLengthEstimation(foc)`estimates of focal length value over a given set of candidates using Kernel voting scheme. Distribution curves are plotted.
  `Foc –` given set of focal length candidates containing complex and real roots

```
Example usage:

clear all; close all;
% retrieve a set of candidates saved in variable foc
load roots;
FocalLengthEstimation(foc);
```

`Test_scaled_rec.m` performs a suggested method for multiview reconstruction on the synthetic data. Various levels of noise can be used.

These examples are included in file ‘Main_implementations.m’ on the enclosed CD.