Visual servoing-based camera control for the da Vinci Surgical System

Cecília Molnár
Antal Bejczy Center for Intelligent Robotics
Óbuda University, Bécsi út 96/b, 1034, Budapest, Hungary
Budapest, Hungary
cecilia.molnar@irob.uni-obuda.hu

Tamás D. Nagy
Doctoral School of Applied Informatics and Applied Mathematics
Antal Bejczy Center for Intelligent Robotics
Óbuda University, Bécsi út 96/b, 1034, Budapest, Hungary
Budapest, Hungary
tamas.daniel.nagy@irob.uni-obuda.hu

Renáta Nagyné Elek
Doctoral School of Applied Informatics and Applied Mathematics
Antal Bejczy Center for Intelligent Robotics
Óbuda University, Bécsi út 96/b, 1034, Budapest, Hungary
Budapest, Hungary
renata.elek@irob.uni-obuda.hu

Tamás Haidegger
University Research and Innovation Center
Óbuda University, Bécsi út 96/b, 1034, Budapest, Hungary
Budapest, Hungary
haidegger@irob.uni-obuda.hu

***Abstract*—Minimally Invasive Surgery (MIS)—which is a very beneficial technique to the patient but can be challenging to the surgeon—includes endoscopic camera handling by an assistant (traditional MIS) or a robotic arm under the control of the operator (Robot-Assisted MIS, RAMIS). Since in the case of RAMIS the endoscopic image is the sole sensory input, it is essential to keep the surgical tools in the field-of-view of the camera for patient safety reasons. Based on the endoscopic images, the movement of the endoscope holder arm can be automated by visual servoing techniques, which can reduce the risk of medical error. In this paper, we propose a marker-based visual servoing technique for automated camera positioning in the case of RAMIS. The method was validated on the research-enhanced da Vinci Surgical System. The implemented method is available at: https://github.com/ABC-iRobotics/irob-saf/tree/visual_servoing**

***Index Terms*—Robot-Assisted Minimally Invasive Surgery, Surgical Robotics, Visual Servoing, Subtask Automation, Open-Source**

I. Introduction

Surgeons are facing numerous challenges in the operating room, such as overwhelming stress, communication and teamwork with the operating staff, late-night shifts and special-need patients. Minimally Invasive Surgery (MIS) was a paradigm change in medicine, where instead of opening the operating area, the intervention is performed through small skin incisions [1]. During MIS, the operating area is visualized by an endoscopic camera image. While MIS has several advantages to the patient (quicker healing time, less pain, smaller scars, etc.), it can be challenging to the surgeon (narrowed endoscopic vision, limited range of motion and vision, low ergonomy, etc.). Robot-Assisted Minimally Invasive Surgery (RAMIS) was introduced to overcome much of these problems; such teleoperated systems provide 3 dimensional vision through a stereoscopic endoscope, offering higher ergonomy, more intuitive instrument control, rescaled motion and some low-level safety functions, such as tremor filtering [2]. The market leader RAMIS system is the da Vinci Surgical System (Intuitive Surgical Inc. Sunnyvale, CA), which received FDA clearance in 2000. Since then, six generations appeared of the robot, following the same control principle [3] (Fig. 1). Due to the fact that RAMIS systems offer various benefits, there are over one million successful RAMIS procedures performed a year [4].

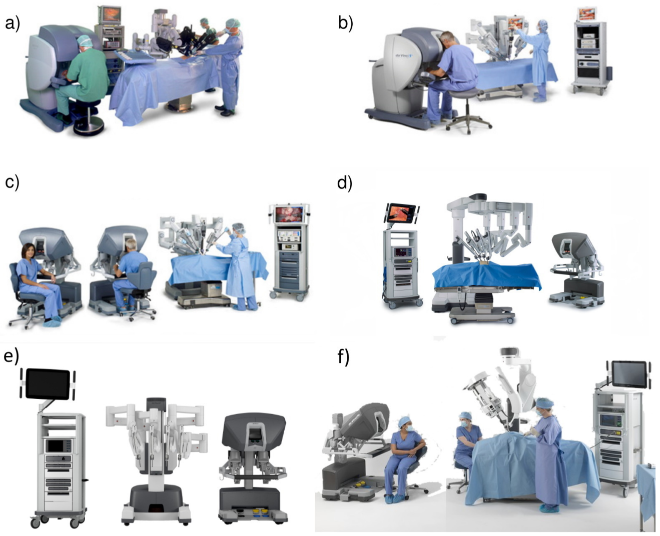

Fig. 1. The 6 generations of the da Vinci Surgical System; a) da Vinci Classic, b) da Vinci S, c) da Vinci Si and d) da Vinci Xi, completed with the most recent e) X and f) SP systems. Image credit: Intuitive Surgical Inc.

Nevertheless, some limitations of RAMIS remains. The endoscopic camera mounted on the Endoscopic Camera Manipulator (ECM) is controlled by the surgeon through foot pedals by default, which presents a cognitive load [5]. It still poses a major risk when the surgical tools are not in the field-of-view of the camera, the tissues can easily be damaged by the tools. The camera is controlled by the surgeon through foot pedals by default. A possible solution for reducing the risk associated with the manual control is the autonomous positioning of the endoscopic camera. Detecting and following the movements of the Patient Side Manipulators (PSMs) can be achieved by eye-in-hand visual servoing algorithms using the frames of the endoscopic camera [6].

In this paper, we propose an open-source software solution for visual servoing for camera control of a da Vinci classic with Da Vinci Research Kit (DVRK) [7]. During the research, the main goal was to extend the iRob Surgical Automation Framework (`irob-saf`) [8] with visual servoing for the ECM. The technique of visual servoing uses visual feedback to control the motion of a robot. The automation of the endoscope's motion can by achieved utilizing visual servoing, by the means of the ECM follows the movements of the PSMs, therefore the surgical tools are always in the field-of view of the camera. Developing the open-source software solution for visual servoing for camera control of a da Vinci classic means creating a more autonomous system for da Vinci, which is considered to be at Level of Autonomy 2 (LoA 2) [4]. At LoA 2, the system is trusted to complete certain tasks or sub-tasks in an autonomous manner, such as suturing or blunt dissection. The visual servoing feature of the proposed framework can assist through various parts of the whole surgery, and its tasks do not involve decision making. In this recent work, we focus mainly on the visual servoing robot control, thus we used a marker-based tool detection. We validated our method on a the DVRK system at Óbuda University.

II. Methods

Our proposed method uses eye-in-hand visual servoing approach; it allows the autonomous movements of the camera by receiving an excepted goal function (expected view). The method uses stereoscopic images as an input, the positioning of the ECM operates in 3D. The tool detection was achieved via ArUco codes (marker-based detection) placed on the PSM [9]. The schematic figure of the setup is presented in Fig. 2.

A. Software Frameworks

a) Robot Operating System: The Robot Operating System[1] (ROS) is a flexible framework for the modular development of robot software [10]. It is a collection of functions, libraries and conventions, aiming to simplify the development of complex and robust robot applications across a wide variety of platforms. It consists of nodes, which are responsible for specific functions. Those communicate with each other via messages though channels called topics. The messages are specific data structures serving various purposes.

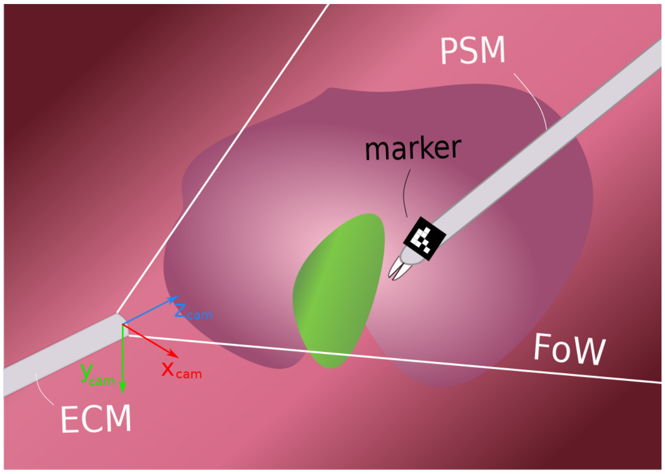

Fig. 2. The setup of the `irob-saf` with visual servoing: The eye-in-hand visual servoing can be executed if the PSMs (detected with markers) are in the field-of-view (FoW) of the ECM.

b) Da Vinci Research Kit: DVRK is a set of open-source, custom-built hardware controllers and software elements to make possible the programming of the attached da Vinci arms, developed for researchers [7]. Amongst other methods the DVRK offers ROS interface to the arms, overriding the original control of the da Vinci; this interface was used to control the ECM.

c) IROB Surgical Automation Framework: The purpose of the `irob-saf`[2] is to facilitate automation of surgical subtasks for the da Vinci Surgical System with the DVRK, offering a modular architecture with built-in functionalities, like parameterizable surgemes, interface to the DVRK or computer vision algorithms and image pipeline from the endoscope [8]. In this research, the goal was to extend the `irob-saf` with depth-inclusive visual servoing for the ECM.

B. Architecture of the `irob-saf` with visual servoing

The architecture of the `irob-saf` with visual servoing is presented in Fig. 3. The nodes of the developed system can be divided into two domains—visual and motion. These two domains are communicating through the main node, called *subtask logic*, containing the high-level implementation of the specific subtask.

The visual domain is basically an image processing pipeline, highly relying on the built-in stereo vision capabilities of ROS[3][4]. On the lowest level, the stereo video stream from the endoscope, captured by a Blackmagic DeckLink frame grabber card[5], is received by a built-in *ArUco detector* node of `irob-saf`. The result of this node is further processed by the node calculates the 3D position of the marker. This 3D position is received by the *subtask logic* node, and used for the manipulation of the endoscope.

[1] https://www.ros.org/

[2] https://github.com/ABC-iRobotics/irob-saf

[3] http://wiki.ros.org/stereo_image_proc

[4] http://wiki.ros.org/stereo_image_proc

[5] https://www.blackmagicdesign.com/products/decklink

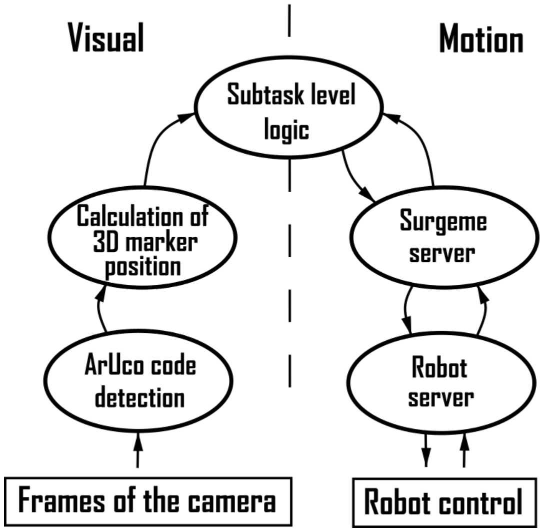

Fig. 3. The hierarchic structure of the `irob-saf` with visual servoing: the visual domain provides the 3D coordinate of the marker from the stereo video stream; the motion domain synthesize the motion and execute it.

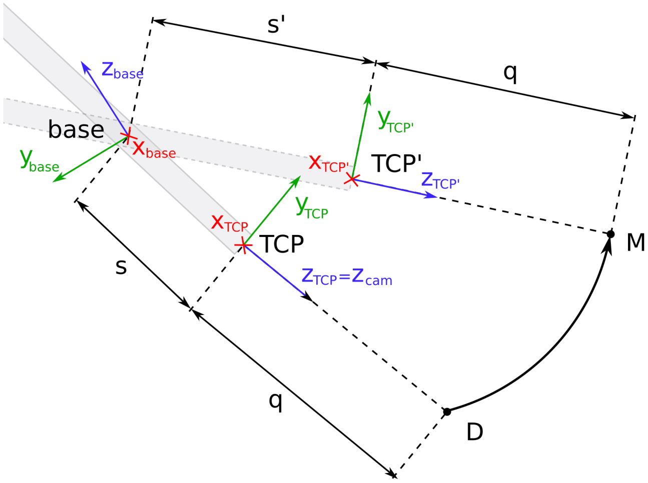

Fig. 4. The concept of following the instrument using visual servoing on the da Vinci surgical System. The TCP' pose of the Endoscopic Camera Manipulator (ECM) to be calculated, so the instrument is seen at the desired position in the camera image.

The motion domain is built in a hierarchic structure; on the top level, the node is called the *surgeme server*, containing the implementation of surgical gestures, such as cutting or grasping. Below the *surgeme server*, the *robot server* node is performing high-level robot control tasks, such as following a trajectory. The *robot server* is connected to the ROS interface of the DVRK, reliable for the low-level control of the arms.

C. Extraction of the Instrument Position

As mentioned above, ROS offers extensive stereo vision support, from stereo camera calibration to the calculation of the 3D point cloud of the scene. Utilizing this infrastructre the 3D position of the ArUco marker is calculated as follows. The image coordinates of the corners of the detected ArUco marker is received alongside the 3D point cloud calculated by ROS. The image coordinates of the marker's center is then calculated from the received corners; these coordinates are used to extract the marker's 3D position from the point cloud using the Point Cloud Library (PCL)[6]. The 3D position of the marker is published to the subtask level node.

D. Robot Control with Visual Servoing

The purpose of the developed visual servoing algorithm is to keep the tracked surgical instrument within the bounds of the endoscopic camera image, while avoiding unnecessary movements of the ECM, as it might disturb the surgeon. Thus, instead of implementing a controller that would keep the instrument in the center of the image by constantly adjusting the ECM pose, the following approach was utilized. The desired position of the instrument (D) and a distance threshold (t) are to be defined in the camera frame (Fig. 2) by the user; the position of the instrument (currently and ArUco marker, M) is tracked by the stereo camera of the ECM (Fig. 4). If the distance of the instrument and the desired position exceeds the threshold t in any direction, the ECM is moved to a new pose (TCP'), where the instrument's position in the camera frame is the same as the desired position; it is in the center of the image again. The pose TCP' is calculated as follows.

The motion of the ECM is restricted in a way so that at the point of insertion (inside the trocar) lateral motion is not possible; it would potentially harm the patient. This point is the so-called Remote Center of Motion (RCM), which is also the origin of the *base* frame of the ECM (Fig. 4). Due to this lateral restriction, the ECM has 4 Degrees of Freedom (DoF): pivot around the RCM, insertion and rotation along the shaft.

The position of points D and M is defined/measured in the coordinate frame of the camera (*cam*, Fig. 2). As the marker is intended to keep near the center of the camera image, in distance q—defined by the surgeon or the assistance—from the camera (Fig. 4), the homogeneous coordinates of desired position of the marker is written as follows:

$$\mathbf{d}_{cam} = \begin{bmatrix} 0 & 0 & q & 1 \end{bmatrix}^T . \quad (1)$$

To calculate the new TCP' pose, the coordinates of those points need to be converted to the TCP frame (Fig. 4). In the case of a 0 degree endoscope, this—by neglecting the displacement of the the camera along axis x—is simply a π rotation around axis x, with homogeneous transformation:

$$\mathbf{T}_{TCP,cam} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (2)$$

Then, the coordinates of the points D and M in the *base* frame are calculated using the position and orientation of

[6] https://pointclouds.org/

TCP, received from the DVRK, converted to homogeneous transformation $\mathbf{T}_{TCP,base}$.

Thanks to the ECM's restricted form of motion, the calculation of the pose TCP' gravely simplifies by conversion to a spherical coordinate system around the origin of the $base$ frame. Defining the spherical coordinates of the points D and M in the $base$ frame would easily result in *inclination* π, as those points are typically located in the vicinity of the elongation of axis z_{base} (Fig. 4). If the inclination is 0 or π, the *azimuthal angle* is arbitrary; that ambiguity is to be avoided in this application. Thus, the frame of the spherical coordinate system (*sph*) is rotated by $\pi/2$ along axis x:

$$\mathbf{T}_{sph,base} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3}$$

The homogeneous coordinates of points D and M in the frame *sph* are then calculated by

$$\mathbf{d}_{sph} = \mathbf{T}_{sph,base} \cdot \mathbf{T}^{-1}_{TCP,base} \cdot \mathbf{T}_{TCP,cam} \cdot \mathbf{d}_{cam} \tag{4}$$

and

$$\mathbf{m}_{sph} = \mathbf{T}_{sph,base} \cdot \mathbf{T}^{-1}_{TCP,base} \cdot \mathbf{T}_{TCP,cam} \cdot \mathbf{m}_{cam} \tag{5}$$

consecutively, where $\mathbf{d}_{cam}$ and $\mathbf{m}_{cam}$ are the homogeneous coordinates in the camera frame. The spherical coordinates r (radial distance), θ (inclination) and ϕ (azimuthal angle) of those points are calculated by the following formulae:

$$r = \sqrt{x^2 + y^2 + z^2}, \tag{6}$$

$$\phi = \arctan2(y, x), \tag{7}$$

$$\theta = \arccos\frac{z}{r}. \tag{8}$$

By knowing the spherical coordinates of the desired and the current tool locations, the difference of those coordinates can be used to get the transformation from TCP to TCP'. The angular differences

$$\Delta\phi = \phi_M - \phi_D; \tag{9}$$

$$\Delta\theta = \theta_D - \theta_M \tag{10}$$

corresponding to the rotation from TCP to TCP', and distance

$$\Delta r = r_M - r_D \tag{11}$$

corresponding to the insertion of the endoscope, are defined. Important to note that the $\Delta\theta$ is intentionally defined with opposite sign, as the the direction of the inclination angle is opposite to the rotation defined by axis x.

From the spherical coordinates it is easy to see that the required length of the ECM's insertion, that ensure the q distance from the tracked instrument, is $s' = s + \Delta r$, where s is the current length of insertion (Fig. 4). The vector for this translational movement is calculated as follows:

$$\mathbf{t}_{TCP} = \begin{bmatrix} 0 & 0 & \Delta r \end{bmatrix}^T; \tag{12}$$

$$\mathbf{t}_{base} = \mathbf{R}_{TCP,base} \cdot \mathbf{t}_{z,TCP}, \tag{13}$$

where $\mathbf{R}_{TCP,base}$ is the rotational part of $\mathbf{T}_{TCP,base}$. The homogeneous matrix for this translation is:

$$\mathbf{T}_r = \begin{bmatrix} \mathbf{I} & \mathbf{t}_{base} \\ \mathbf{0} & 1 \end{bmatrix}. \tag{14}$$

As the azimuthal angle defines a rotation around axis z, and the inclination a rotation around axis x, the required rotations from the differences are written:

$$\mathbf{T}_{az} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\Delta\theta) & -\sin(\Delta\theta) & 0 \\ 0 & \sin(\Delta\theta) & \cos(\Delta\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{15}$$

Similarly, as inclination defines the rotation along axis z:

$$\mathbf{T}_{inc} = \begin{bmatrix} \cos(\Delta\phi) & -\sin(\Delta\phi) & 0 & 0 \\ \sin(\Delta\phi) & \cos(\Delta\phi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{16}$$

Finally, using these transformations the desired pose of the ECM TCP' is calculated:

$$\mathbf{T}_{TCP',base} = \mathbf{T}^{-1}_{sph,base} \cdot \mathbf{T}_{inc} \cdot \mathbf{T}_{az} \cdot \mathbf{T}_{sph,base} \cdot \mathbf{T}_r \cdot \mathbf{T}_{TCP,base}. \tag{17}$$

Using the desired pose of the ECM a Cartesian trajectory is generated and being sent to the controller to move the ECM into that pose; positions are generated by linear interpolation, the orientation by spherical linear interpolation (Slerp) [11], offered by the Eigen C++ library[7].

III. Results

To validate the proposed visual servoing method, experimental scenarios were set up, where the measurement of the accuracy were aimed. The tests on the da Vinci Surgical System were done in DVRK teleoperated mode with a phantom representing a gallbladder and its environment. During the test setups, the arms are controlled from the master side by an operator, except the ECM, which was moved autonomously by the proposed method. The anatomically correct surgical phantom (originally created for modeling laparoscopic cholecystectomy) provided the background for the tests, and an ArUco code was placed on the moving PSM (Fig. 5). To determine the accuracy of the system, the ArUco marked PSM was moved in x, y and z directions by the operator. The Cartesian position of the ECM alongside the joint angles

[7] http://eigen.tuxfamily.org

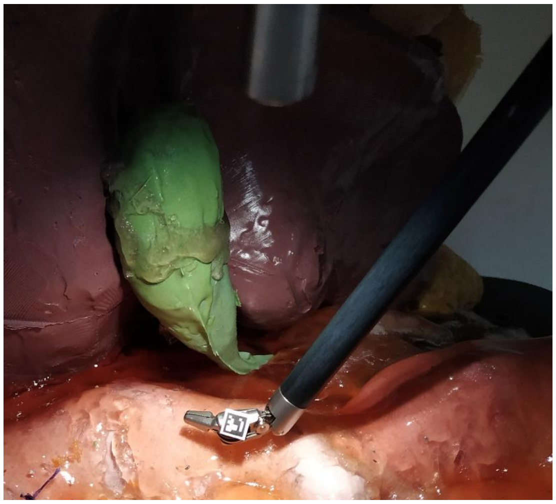

Fig. 5. Testing the proposed visual servoing method on the da Vinci Research Kit. An ArUco code was fixed on a surgical tool for instrument tracking, and the endoscope followed the tool's displacements. The tests were done in a surgical phantom environment.

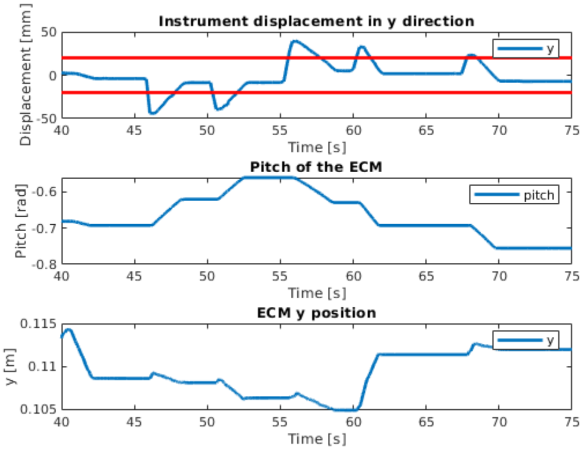

Fig. 7. One-marker setup for y axis experiment. The red horizontal lines are the given threshold value for the displacement length.

related to *yaw*, *pitch* and *insertion* of the endoscope were tracked, then shown in the graphs to easily determine if the created system functions properly. In the experiments, the endoscopic camera of the da Vinci was used with the original 640 × 480 pixels resolution camera of the da Vinci Classic.

First, an experiment of one instrument, with an attached ArUco marker moving in the x axis of the coordinate frame of the camera was performed. The results of the distance from the desired position in the x direction, the *yaw* angle and x component of the ECM's Cartesian position are shown in Fig. 6.

Similarly, in the second experiment the marked instrument was moved along the y axis of the coordinate frame of the camera, while additionally to the marker's distance from the desired pose in this direction the *pitch* and the y component of the ECM's Cartesian position are recorded (Fig. 7).

In the third experiment, the instrument was moved along the z axis of the coordinate frame of the camera; apart from the z component of the distance of the marker from the desired position and z component of the position of the ECM, the value of the *insertion* joint were captured (Fig. 8).

In Fig. 6, 7 and 8, the distance threshold value of $\pm 20\ mm$ is marked with red horizontal lines. The two threshold lines determine a limited area corresponding to the ideal point, where the ECM does not move. Outside this area, the ECM performs its corrections to reach the desired point determined by the thresholds as shown in those figures. The desired values along both axes x and y were 0 (corresponding to the center of the image), and $30\ mm$ along z axis (corresponding to $30\ mm$ distance from the camera).

The data of three graphs above are shown together, along-

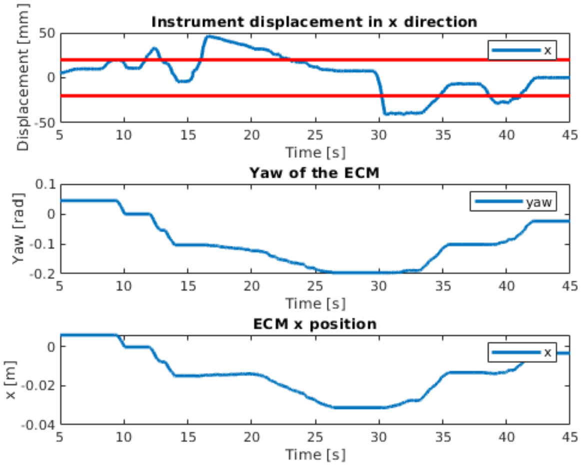

Fig. 6. One-marker setup for x axis experimentn. The red horizontal lines are the given threshold value for the displacement length.

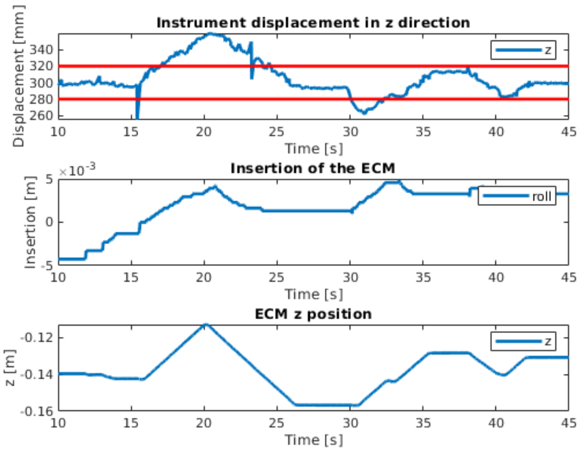

Fig. 8. One-marker setup for z axis experiment. The red horizontal lines are the given threshold value for the displacement length.

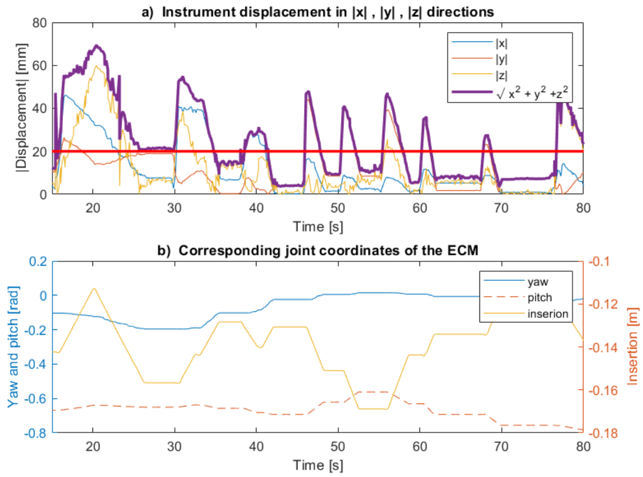

Fig. 9. One-marker setup for all directions shown in one graph. The red horizontal line is the given absolute threshold value for the displacement length. The purple line is the Euclidean distance of the marker from the desired point.

side the Euclidean distance from the desired position in Fig. 9, where the relationship between the directions of the marker's displacement and the changing of corresponding joint coordinates can be observed. In Fig. 9a, the absolute value of the instrument displacement in x, y, z directions are shown with the Euclidean distance of the marker from the desired point. In Fig. 9b, corresponding joint coordinates of the ECM *(yaw, pitch, insertion)* are shown. It the instrument displacement in x direction is higher than the set threshold, the corresponding joint coordinate of the ECM, the yaw will alter. Similarly, the pitch corrects the instrument displacement in y direction, and the instertion corrects the instrument displacement in z direction. If the movements in the experiments were complex, two or more joints were moving in the given interval.

IV. Limitations and Further Works

This recent study focused mainly on robot control, however, advanced computer vision-based, marker-less approaches are expected in the clinical practice. Instrument segmentation in RAMIS is a well-studied area, most of the recently proposed methods are Deep Neural Network-based solutions [12], [13]. In our future work, we intend to add existing marker-less instrument segmentation implementations. Another limitation of the system is the one marker-based approach; in the medical practice, two or more instruments are often used. To keep two instruments in the visual scene, we have to extend our method to handle this, especially in the case when the two tools are moving to different directions. In this work, we used a 0 degree endoscope; however, 30 degree endoscopes are often used in RAMIS. In our future work the case of different endoscope geometries will also be considered.

V. Conclusion

In this work, we proposed a marker-based visual servoing method for Robot-Assisted Minimally Invasive Surgery to automate camera moving, which can help the surgeon focusing only on the instrument control in DVRK enhanced research mode. We tested our method on the da Vinci Surgical System. The usage of visual markers ensured the elimination of a significant fraction of vision-related errors, thus during the development and validation the main focus could remain on the robot control aspect of the visual servoing problem. The outcome showed that the framework can be considered a reliable base for future work. Further development needs to be done to extend the work, such as advanced computer vision-based tool detection and multiple instruments tracking.

Acknowledgment

R. Nagyné Elek, T. D. Nagy and T. Haidegger are supported through the New National Excellence Program of the Ministry of Human Capacities. This work was partially supported by ACMIT (Austrian Center for Medical Innovation and Technology), which is funded within the scope of the COMET (Competence Centers for Excellent Technologies) program of the Austrian Government. T. Haidegger is a Bolyai Fellow of the Hungarian Academy of Sciences.

References

[1] L. J. Freeman, "Method for minimally invasive surgery in the digestive system," Patent, 2003.

[2] Á. Takács, D. Á. Nagy, I. J. Rudas, and T. Haidegger, "Origins of Surgical Robotics: From Space to the Operating Room," *Acta Polytechnica Hungarica*, vol. 13, no. 1, pp. 13–30, 2016.

[3] K. Miller and M. Curet, "Intuitive Surgical: An overview," in *Robotic-Assisted Minimally Invasive Surgery*. Springer, 2019, pp. 3–11.

[4] T. Haidegger, "Autonomy for Surgical Robots: Concepts and Paradigms," *IEEE Trans. on Medical Robotics and Bionics*, vol. 1, no. 2, pp. 65–76, 2019.

[5] R. Nagyné Elek and T. Haidegger, "Robot-Assisted Minimally Invasive Surgical Skill Assessment—Manual and Automated Platforms," *Acta Polytechnica Hungarica, Special Issue on Platforms for Medical Robotics Research*, vol. 16, no. 8, pp. 141–169, Sep. 2019.

[6] S. Hutchinson, G. Hager, and P. Corke, "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, Oct. 1996.

[7] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, "An open-source research kit for the da Vinci® Surgical System," in *Proc. of the IEEE International Conference on Robotics and Automation*, Hong Kong, 2014, pp. 6434–6439.

[8] T. D. Nagy and T. Haidegger, "A DVRK-based Framework for Surgical Subtask Automation," *Acta Polytechnica Hungarica, Special Issue on Platforms for Medical Robotics Research*, vol. 16, no. 8, pp. 61–78, 2019.

[9] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[10] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: An open-source Robot Operating System," in *Proc. of the ICRA Workshop on Open Source Software*, vol. 3, Kobe, Japan, 2009.

[11] K. Shoemake, "Animating Rotation with Quaternion Curves," *ACM Transactions on Graphics*, vol. 19, no. 3, pp. 245–254, 1985.

[12] T. Mikada, T. Kanno, T. Kawase, T. Miyazaki, and K. Kawashima, "Three-dimensional posture estimation of robot forceps using endoscope with convolutional neural network," *The International Journal of Medical Robotics and Computer Assisted Surgery*, p. e2062, Jan. 2020.

[13] J. Zhang and X. Gao, "Object extraction via deep learning-based marker-free tracking framework of surgical instruments for laparoscope-holder robots," *International Journal of Computer Assisted Radiology and Surgery*, Jun. 2020.