A DVRK-based Framework for Surgical Subtask Automation

Tamás D. Nagy[1] **and Tamás Haidegger**[1,2]

[1]Antal Bejczy Center for Intelligent Robotics, Óbuda University, Bécsi út 96/b, 1034, Budapest, Hungary, {tamas.daniel.nagy, tamas.haidegger}@irob.uni-obuda.hu
[2]Austrian Center for Medical Innovation and Technology (ACMIT), Viktor-Kaplan-Straße 2/1, 2700, Wiener Neustadt, Austria, tamas.haidegger@acmit.at

Abstract: Robotic assistance is becoming a standard in Minimally Invasive Surgery. Despite its clinical benefits and technical potential, surgeons still have to perform manually a number of monotonous and time-consuming surgical subtasks, like knot-tying or blunt dissection. Many believe that the next bold step in the advancement of robotic surgery is the automation of such subtasks. Partial automation can reduce the cognitive load on surgeons, and support them in paying more attention to the critical elements of the surgical workflow. Our aim was to develop a software framework to ease and hasten the automation of surgical subtasks. This framework was built alongside the Da Vinci Research Kit (DVRK), while it can be ported onto other robotic platforms, since it is based on the Robot Operating System (ROS). The software includes both stereo vision-based and hierarchical motion planning, with a wide palette of often used surgical gestures—such as grasping, cutting or soft tissue manipulation—as building blocks to support the high-level implementation of autonomous surgical subtask execution routines. This open-source surgical automation framework—named `irob-saf`*—is available at* `https://github.com/ABC-iRobotics/irob-saf`.

Keywords: robot-assisted minimally invasive surgery; surgical robotics; subtask automation; open-source platform

1 Introduction

In the last few decades, the headway of Minimally Invasive Surgery (MIS) had a significant influence on surgical practice. Contrary to traditional procedures performed using large incisions, MIS is executed through few-centimeter-wide ports, using so-called laparoscopic instruments, while the surgeon observes the area of operation via endoscopic camera stream. The smaller incisions required by the MIS technique have a number of benefits for both the patient and the hospital; causing less trauma and lowering the risk of complications. MIS also shortens recovery time and hospital stay. Nevertheless, this technique presents

serious cognitive and ergonomic challenges to the surgeons, like the limited range of motion, or weary positions.

The next step in the evolution of MIS was the introduction of teleoperated master–slave surgical systems. The fundamental idea of teleoperated surgery originates from space research [1]; the patient—in this case an astronaut—was to be treated by a slave device controlled by a remote surgeon, sitting at a master device on Earth. On the slave side, robot arms hold laparoscopic instruments, and copy the movement of the surgeon at the master console. To ensure visual feedback, the slave side device is equipped with an endoscopic camera, whose video stream is also sent to the master device and displayed to the surgeon.

Primarily due to the issues caused by time delay and system complexity, the idea of long distance telesurgery has not become a daily practice, and stalled in the state of research and pilots. Nevertheless, communication latency can be reduced to an insignificant level when the master and the slave devices are close to each other. Commercial Robot-Assisted Minimally Invasive Surgery (RAMIS) was born along this idea, where the master and the slave devices are in the same room. These systems are able to ease the fatigue of the surgeon, since they can operate in a comfortable, seated position, in a more ergonomic environment. Moreover, the motion of the surgeon is scalable, which means that the most delicate and fine maneuvers can be controlled by relatively large hand movements.

Undoubtedly, the most successful RAMIS device is the da Vinci Surgical System (Intuitive Surgical Inc., Sunnyvale, CA). Its 1st generation was cleared by the U. S. Food and Drug Administration in 2000, and became a commonly used device in a few years. Today, the 4th generation—da Vinci Xi—is available, while the more affordable alternative X, and the Single Port solution (SP) is also in the product portfolio of the company (Fig. 1). More than 5500 da Vinci units are installed worldwide that performed around 1 million procedures last year [1].

2 Da Vinci Research Kit

In the mid-2010s, the 1st generation of the robot (the da Vinci classic) was sent to retirement, since those were impractical to be serviced and supplied anymore. Nevertheless, the retired da Vincis were still functional, and could be well used in more failure-tolerant applications, like research. The development of the Da Vinci Research Kit (DVRK) was started at the Johns Hopkins University, and in a few years, an active community has gathered with more than 30 setups worldwide [2].

The DVRK consists of open-source, custom-built hardware controllers and software elements to make possible the programming of the attached da Vinci arms. The controllers of DVRK are built from two custom boards, an IEEE-1394 FPGA board and a Quad Linear Amplifier (QLA); these provide the computational power and low latency communication required for the low-level high-frequency robot control. The controller boxes are interfaced to PC using IEEE 1394a

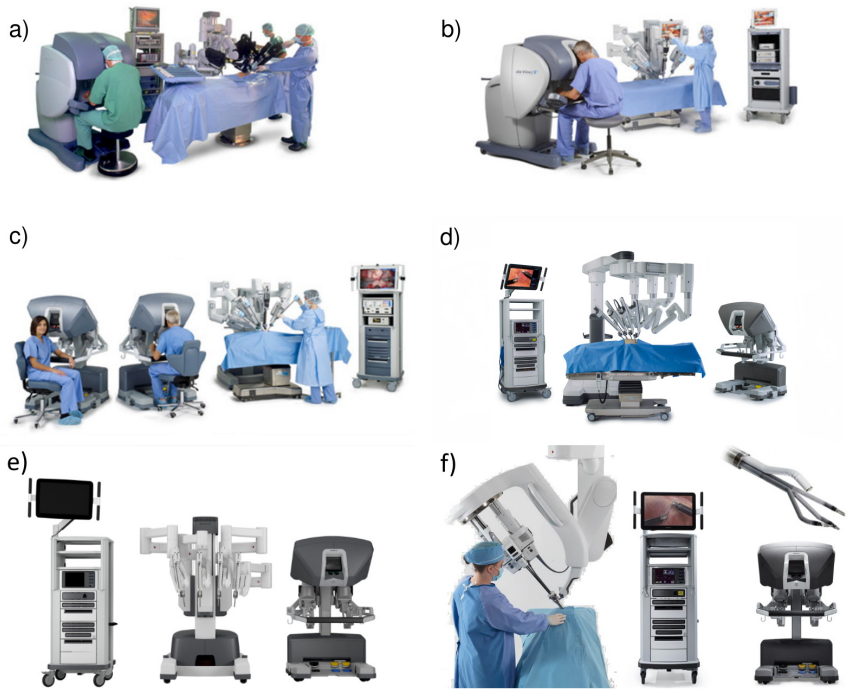

Figure 1
The 4 generations of the da Vincy Surgical System; a) da Vinci Classic, b) da Vinci S, c) da Vinci Si and d) da Vinci Xi, completed with the most recent e) X and f) SP systems. Image credit: Intuitive Surgical Inc.

(FireWire); one of the controllers have to be connected directly to the PC, then this controller is able to manage the communication of further controllers in a daisy chain. On the PC side, the open-source *cisst* libraries [3] are for the mid-level control and FireWire communication, built by *Catkin* build system. These libraries themselves offer the functionality to the programming of the robot, however, a ROS interface was also implemented; more than half of the research institutes use the latter for the programming of the da Vinci [4].

ROS is used commonly in robotics, mainly in the field of research, and most of the research centers working with the DVRK, or the RAVEN platform [4], rely on ROS. Beyond the compatibility with the mentioned platforms, ROS is quite powerful, offering built-in solutions for a number of problems e.g., accessing sensory data, calibrating stereo-cameras, and enabling highly modular development.

3 Subtask Automation in Surgery

Many believe that the next step in the technical development of RAMIS is going to be partial (or conditional) automation [5]. The surgical workflow of RAS procedures often contains time-consuming and monotonous elements—so-called surgical subtasks—like suturing, knot-tying, or blunt dissection. The automation of these subtasks can reduce the fatigue and the cognitive load on the surgeon, who can hence pay more attention to the more critical parts of the intervention [6].

As the development of the technological background in the last couple of years offers a rising potential, like deep learning or mechatronics, the automation of surgical subtasks became a prevailing topic in the research of surgical robotics. A number of autonomous surgical subtasks are already implemented, or being currently developed by various research groups. A list of relevant subtasks in the research of surgical automation is compiled in Table 1.

Table 1
List of surgical subtasks from the aspect of suitability for partial automation.

Subtask	**Sensor integration**	**Experimental environment**	**Complexity**	**Clinical relevance**	**Ref.**
shape cutting	stereo camera	gauze patch, FRS Dome[1]	medium	high	[7]
suturing	stereo camera	silicone, foam, FRS Dome	high	high	[8]
ligation	–	special phantom	medium	high	[9]
palpation	force sensor	special silicone phantom, FRS Dome	medium	medium	[10] [11]
tumor palpation and resection	force sensor	special silicone phantom, FRS Dome	high	medium	[12]
debridement	stereo camera	tiny objects	medium	high	[13] [14]
suction and debridement	–	special phantom	medium	high	[15]
bowel anastomosis	3D camera	porcine bowel	high	high	[16]

[1] Florida Hospital Nicholson Center, Celebration, FL

blunt dissection	stereo camera	sandwich-like silicone phantom	medium	high	[17] [18]
tissue retraction	stereo camera	silicone phantom	low	high	[19]
peg transfer	stereo camera	training phantom	medium	low	[20]

The automation of a number of subtasks are currently under active research, such as different aspects of suturing, soft tissue cutting, debridement, palpation or blunt dissection, employing techniques like learning-by-observation, motion decomposition and state machines [7–20].

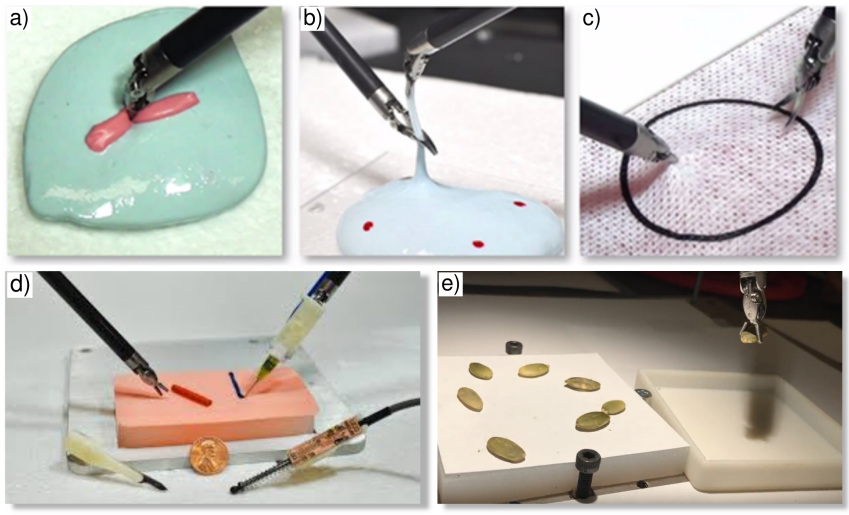

Figure 2
Recently automated surgical subtasks. a-c) Multilateral cutting, d) tumor palpation and e) resection, debridement. Image credit: [7, 12, 14].

All of the mentioned surgical subtasks are to be performed on soft tissue, in a highly deformable environment. In contrast to subtasks involving hard tissue, like bone cutting, where the target organ can be fixed and registered with the surgical device via a navigation system, soft tissue presents new challenges from the aspect of automation, as the robot has to operate in unpredictable environment. Probably the biggest challenge is the development of perception algorithms; it is not trivial how the information, needed for the execution of the current subtask can be extracted from the surrounding soft, specular environment. Despite the fact that working implementations could be found e.g., on instrument segmentation/pose estimation [21, 22] or organ segmentation and 3D reconstruction [23–25], autonomous navigation inside the patient's body still presents a

huge challenge being under intensive research. As of today, shared control is a more viable option for these clinical routines [26, 27]. Furthermore, the generation of required motion patterns and the design of control methods for the manipulation of unknown soft tissues is also problematic [28].

Our aim was to develop an open-source framework to support such development projects; to provide software packages that contains already implemented basic functionalities, eventually becoming universal building blocks in surgical subtask automation [20]. The architecture of this software package—the iRob Surgical Automation Framework, or `irob-saf`—is presented herein.

4 Materials and Methods

One of the fundamental tasks in the development of this surgical automation framework was the hierarchical decomposition of surgical motion patterns. The workflow of surgical interventions, as well as the motion of the surgeon, can be decomposed into elements on different levels of granularity [29–31], similar to behavior trees [32]. In the literature, there are several different definitions of some granularity levels, nevertheless, no consistent definition can be found for the whole domain. To decompose surgical motion and implement partial automation, it is necessary to define these levels as precisely as possible. For that manner, we defined the levels of granularity—according to the current state of research—as follows (Fig. 3):

1. **Operation:** The entire invasive part of the surgical procedure.
2. **Task:** Well delimited surgical activity with a given high-level target/goal to achieve.
3. **Subtask:** Circumscribed activity segments that accomplish specific minor landmarks in completing the surgical task.
4. **Surgeme:** An atomic unit of intentional surgical activity resulting in a perceivable and meaningful outcome.
5. **Motion primitive:** General elements of motion patterns, that can be directly translated into robot commands.

In most studies, the granularity level chosen for surgical automation is the level of subtasks (Table 1). The execution of those subtasks usually leads to the accomplishment of a specific milestone, which is in line with the term of partial automation. Subtasks can be further divided into surgemes, which are universal to different subtasks. Thus, from the viewpoint of automation, different subtasks can be built of a set of universal surgemes. Those thoughts lead to the assembly of a motion library (*irob-saf*), containing a set of universal surgeme implementations.

To build this motion library, a number of surgical subtasks had to be decomposed into a set of universal surgemes. For that purpose, several features and events

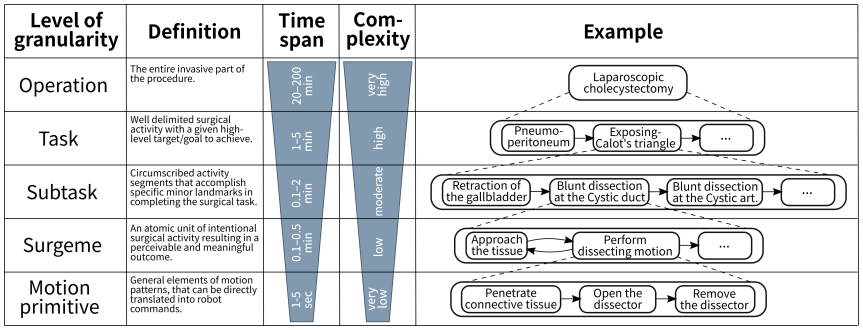

Figure 3
Overview of surgical motion's granularity levels. Mapping of an example, Laparoscopic Cholecystectomy procedure onto different granularity levels [17, 20].

were defined that separates subsequent surgemes from one another. A prime one is the overall shape of motion; this distinguishes for example the cutting from free navigation. Another important feature is the presence of tissue interaction during the surgeme; the instrument can move freely in the abdomen, it can grasp a loose piece of tissue, or even manipulate a tissue layer anchored to the anatomy. If the type of tissue interaction changes during the subtask execution, it will surely means the transition to another surgeme. The final aspect of decomposition was the instrument required to be used during the procedure, e.g., a grasping surgeme might not be performed using scissors, and a cutting might not be done using grasping tools.

5 The Architecture of the Framework

The ROS platform—used widely in robotics—offers solutions to build modular, reusable software on a large scale. A ROS-based architecture consists of so-called *nodes*, intercommunicating with each others over channels of three types:

- Topic: continuous data streaming
- Service: request–response type communication with blocking behavior, has benefits for e.g. requesting calculations
- Action: request–response type communication with non-blocking behavior, useful for environmental interactions.

Due to its benefits, the `irob-saf` framework was completely built on ROS, and tailored to be usable alongside the DVRK. However, due to the implemented ROS interface, the framework is easily portable to other platforms. A system, performing a surgical subtask autonomously, can be assembled from the nodes of `irob-saf` based on the principle shown in Fig. 4. Sensors and perception algorithms, directed by ROS nodes, are used for the purpose of the measurement

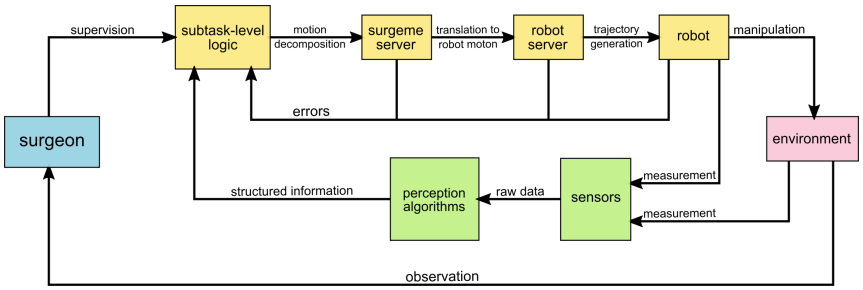

Figure 4
The control scheme of partial automation offered by the framework. Perception nodes gather information from the environment. The subtasks-level logic contains the whole workflow of the subtask, processes the incoming information, and also communicates with the surgeon. This node also sends commands to the hierarchical set of nodes, appointing the surgemes to be executed. The generated motion is executed by the robot under permanent monitoring of the surgeon.

and estimation of the properties of the environment. The information gained, including errors eventually, are all channeled into the subtask-level logic node. This node is responsible for the processing of the information regarding the environment, and the commands originating from the surgeon. Additionally, the surgical workflow is coded in this node; its elements are translated into surgemes and sent to the surgeme server in the form of ROS actions. Propagating down from the surgeme server, the robot motion is generated by a hierarchical network of nodes, then sent to the robot (the ROS nodes from DVRK). Important to note, due to the principle of partial automation, the permanent monitoring of the surgeon is essential during the execution of the subtask.

Camera image is one of the most important sources of information in the automation of RAMIS. The usage of the endoscopic camera image is undoubtedly the most obvious choice, since it does not require the placement of any additional instrument into the already crowded operating room. Nevertheless, in `irob-saf`, the video stream—preferably stereo—can be provided by a wide range of cameras as long as it is interfaced with a ROS topic. Examples interfaces for USB webcameras and the stereo endoscope of the da Vinci are implemented in the framework. The calibration of the cameras—either mono or stereo—is performed by the built-in, easy-to-use camera calibration tool of the ROS environment, using a checkerboard pattern [33]. Furthermore, the basic stereo image processing algorithms, like disparity map calculation or the generation of the 3D point cloud are also performed with one of the built-in libraries of ROS [34].

The algorithms usable for perception are out of scope of the current work, however, the framework offers a pre-built infrastructure to run those with the required input and output channels. These algorithms can be built using C++, Python, or even MATLAB. To ease development, the framework contains examples such as the detection of ChArUco markers [35]. It is important to note that further sensor

modalities, like force sensors, or RGB-D cameras can also be easily added to the existing infrastructure.

The arms of the da Vinci surgical robot are interfaced with the framework by high-level robot control nodes, one node per arm. These nodes are responsible for executing the trajectories generated by higher level nodes, while checking for errors originating from the robot. The trajectories are sent through ROS actions instead of topics, which are more favorable in environmental interaction scenarios. ROS actions makes it possible for the higher-level nodes to do further work during action execution, e.g., monitoring the environment, or sending actions to other nodes. Moreover, actions provide the ability to send feedback and the result of the action, or preempt the action with another, if any environmental change makes it necessary, e.g. the location of the target was changed during execution.

The framework also offers solution for hand–eye calibration; namely the coordinate systems of the arms can be registered to the camera coordinate frame, that makes possible the generation of the robot motion relative to the camera. Visual markers attached to the instruments are used to estimate the tool positions based on the stereo camera stream. The hand–eye calibration can be performed using a MATLAB script, that records tool positions in the robot coordinate frame (received from DVRK through ROS) and in the camera coordinate frame (estimated using the visual markers) simultaneously, in manually set positions. Based on the recorded positions, the optimal rigid transformation is calculated between the two coordinate systems [36–38]. The registration is then saved to a yaml file ("YAML Ain't Markup Language"), that is loaded by the corresponding high-level robot control node, which thus able to receive position commands in the camera coordinate frame from the higher-level nodes of the system.

These high-level robot control nodes are robot-specific, but their interface to the other nodes of the framework is universal. This means that the usage of another type of robot arm requires only the implementation of the high-level robot control node itself.

The surgical motion library, mentioned in Section 4, containing the implementation of universal surgemes, can be found in the package `irob_motion` of the framework. This surgeme library offering surgemes as parameterizable ROS actions, such as: *grasp*, *cut*, *place object*, *release object*, *navigate*, *dissect* and *manipulate tissue*. The implemented surgemes are able to do the necessary safety checks, e.g., the proper instrument is used for the current surgeme. Further surgemes can be implemented based on the existing ones, and then added to the library.

The whole architecture is controlled by a subtask-level logic node. This node is subtask specific, an individual node needs to be implemented for each different surgical subtask. Here is where the information from the perception nodes is received and processed; all the errors, exceptions of the system and user (surgeon) interactions are also channeled; and the surgeme level motion commands are generated. Subtask-level logic nodes are designed to contain and perform the specific workflow of the current subtask. The framework offers skeletons and

also examples how to implement such nodes for the specific surgical subtask. At this level, behavior trees would offer a very structured representation of surgical knowledge and workflow [32], and it is planned to utilize this model in the future development of the framework.

6 Examples

The usage of the framework is explained through two examples on the automation of subtasks. We decided to implement subtasks that require simpler perception methods; those algorithms are out of scope of the current work. The automation of a training exercise and an actual surgical exercise is presented in the following.

6.1 Automating peg transfer

The first example for automation is a RAS training exercise, peg transfer. During this exercise, small tubes have to be placed from one peg to another, to enhance the visuomotor skills of the surgeon (Fig. 5). This exercise is simple enough to present how an autonomous subtask execution can be built using our framework. One and two armed solutions were implemented of the peg transfer exercise. During the one armed execution, the tubes are simply grasped and placed on another peg by one PSM arm. In the two armed version, the tube is grasped and lifted up from the peg from one PSM arm, then transfered to another PSM, that places it on the target peg.

The position of the training board was estimated by the stereo camera stream of the built-in endoscope of the da Vinci. The video stream was captured by a DeckLink Blackmagic (Blackmagic Design Pty. Ltd., Port Melbourne, VIC) card, and forwarded to ROS using GStreamer [39]. The cameras were calibrated using the ROS built-in `camera_calibration` package. The board was marked by ArUco or ChArUco markers, that can be detected robustly by the camera, and can be used to estimate the board's position [35]. To start the nodes for computer vision, the launching of two launch files from the `irob_vision_support` package is necessary:

- `cam_blackmagic_raw.launch`: starts the node for streaming the camera image from one of the da Vinci's cameras
- `charuco_detector.launch`: for the pose estimation of the peg transfer board based on a ChArUco marker.

The nodes responsible for the generation and execution of surgical motion are operating at 4 different levels of hierarchy. The uppermost level is the level of subtasks, with nodes of the `irob_subtask_logic` package. This level is built on a single node, that contains the workflow of the subtasks, receives the pose estimation of the peg transfer board, and chooses the surgemes for execution. The execution of surgemes is requested using ROS actions, that is sent to the proper node in the lower level. The second level of hierarchy contains the im-

plementation of the universal surgemes. At this level, one surgeme server node is launched for each arm operating, receiving ROS actions from the subtask level, and sending ones to the lower, third level. This third level is responsible for the high level control of the arms, and consists of robot server nodes; one such node is responsible for the handling of one arm. These nodes accept ROS action commands for robot movements, and are also connected to the appropriate DVRK node at the fourth, lowermost level to execute the requested movements.

While the nodes of the three lower levels are universal for different subtasks, the uppermost, subtask-level logic node is unique. This node contains the workflow, basically a sequence of surgemes to execute, however, in case of more complex subtasks, a state machine implementation can be useful as well. The motion—both in case of one and two armed solutions—is composed of only three surgemes: *grasp*, *place* and *release* (Fig. 6). All surgemes of the framework including these three, are built of two motion primitives: spatial navigation of the instrument's endpoint, and the movement of the instrument's jaws. These motion primitives can be described well by only a few parameters, and based on the given parameters, the robot trajectories can be easily generated. These three surgemes are built up as follows:

- Grasp:
    1. navigate to approach position (waypoints can be added)
    2. navigate to grasp position
    3. close jaws
- Place:
    1. navigate to approach position (waypoints can be added)
    2. navigate to place position
- Release:
    1. open jaws
    2. navigate to leave position

The execution of those surgemes is requested by sending parameterized actions for to the surgeme server representing the chosen arm. The parameters of those surgeme action requests are calculated by the measured or estimated properties of the environment, received from the computer vision module. Such parameters can be the size of the object to grasp, the compression rate during grasping, or the approach and grasp position of the instrument endpoint.

This hierarchy can be assembled by launching the following instances, in case of two armed execution:

- `peg_transfer_dual.launch` from package `irob_subtask_logic`
- `surgeme_server.launch` from package `irob_motion`, in two instances, parameterized for each arms

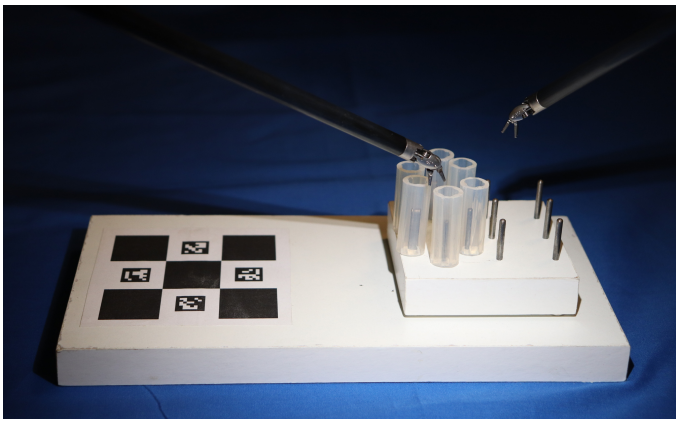

Figure 5
The setup for the peg transfer exercise. The board is marked using a ChArUco marker for image-based pose estimation.

- `dvrk_server` from package `irob_robot`, also in two instances, parametereized for each arms
- DVRK console, with the arms to be operated

The performance of the autonomous execution was compared to humans in the case of the standard exercise, where six tubes are to be placed from the original pegs to another six pegs using two arms. The completion times of six subjects, with minimal expertise in the usage of the da Vinci system were measured after a short practice period. It turned out that the average time needed for the task was 71.4 seconds in the case of novice users, while the automatic agent's performance was 64.0 seconds. However, the speed of the execution could be further increased, at higher speed we found the accuracy of the arms position control started to decrease.

6.2 Automating blunt dissection

Another subtask example implemented using the framework was blunt dissection. Blunt dissection is a common subtask in MIS, usually used to separate loosely connected layers of tissue without damaging sensitive anatomical structures, like nerves or blood vessels. During this subtask, to ensure that none of those sensitive structures get damaged, no sharp instruments are used, the layers are separated by gentle opening movements of the forceps' jaws. This subtask is more relevant from the aspect of surgery, and still simple enough to be automated using simple perception algorithms. The details of this subtask automation were presented in [17].

The development and testing of this algorithm was performed using a silicone phantom consisting of two harder layers of silicone connected with a softer, destructible silicone layer. In our test environment, two calibrated web cameras

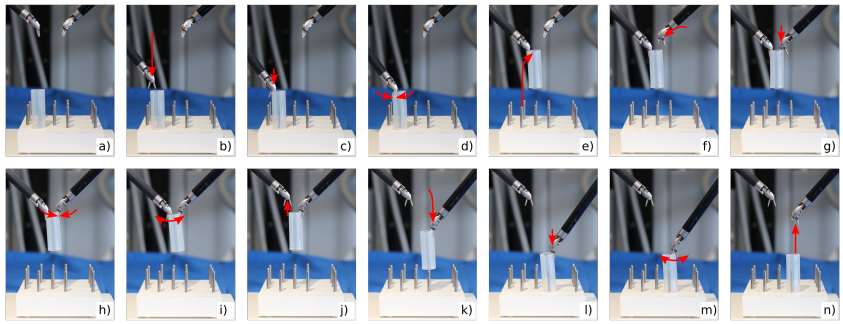

Figure 6
The workflow used in the automation of two-armed peg transfer. a) Setup before starting peg transfer. b–d) Left arm grasps the object. e) The object is lifted to the passing location. f–h) The object is grasped by the right arm. i–j) The object is released by the left arm. k–l) The object is placed on the target peg. m–n) The object is released by the right arm.

were utilized, with fixed focal length, attached onto a stable frame to provide the stereo image feed. The detection of the dissection profile relies on the depth map of the camera scene, calculated from the distance of each corresponding point pair on the rectified stereo pair.

The process presented in Fig. 7 is initiated by manually selecting a starting and an end point of the blunt dissection line. The precise dissection profile, where the dissection will be performed, is selected autonomously, by searching for the local minima of depth in the environment of the points of the manually selected dissection line (Fig. 7). The accuracy of the dissection line detection is further increased using Hampel filter to remove outliers. To ensure to progress evenly inward between the tissues, the point with the lowest depth of the dissection profile is used for the location of the next dissection movement.

As the subtask-level logic node receives the points of the dissection profile, so-called *dissect* surgemes are performed by the arm of the DVRK controlled da Vinci, consisting of the following primitives:

- **Dissect:**
  1. navigate to the point of dissection (Fig. 8/a)
  2. slowly penetrate the tissue (Fig. 8/b)
  3. open the jaws to separate layers (Fig. 8/c)
  4. pull out the instrument in an open position (Fig. 8/d)

The system performing blunt dissection autonomously can be assembled a similar way as the one for peg transfer. In this case, only one arm is required, and the computer vision is implemented in MATLAB. The USB stereo camera pair can be launched by `stereo_cam_usb.launch` of the `irob_vision_support` package.

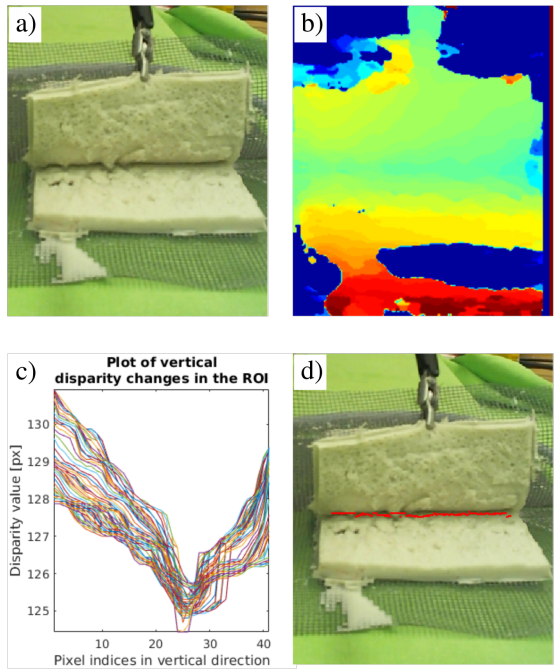

Figure 7
Method for blunt dissection automation via computer vision. a) Image of blunt dissection phantom; b) disparity map of the field of view (color represents the points' distances from the camera); c) plot of disparity changes in vertical direction; d) blunt dissection profile from the local minima of the disparity map. Image credit: [17].

The accuracy of the system was measured during 10 test cases: the average accuracy was 2.2 mm with a standard deviation of 0.5 mm in the camera views plane. In the depth axis—perpendicular to the camera plane—the 1 mm accuracy with standard deviation of 0.6 mm was measured. The overall performance of the autonomous blunt dissection algorithm was evaluated on the silicone-based custom designed phantom, by performing single dissections on 25 different locations. 21 of the 25 attempts succeeded; in 4 cases the dissection profile was missed by a maximum of 3 mm [17].

Discussion and conclusions

An open-source, ROS-based software package was presented, which aims to ease surgical subtask automation research. This framework interfaces sensory inputs, perception algorithms and robots, and contains a surgeme-level motion library. The whole system can be controlled by a subtask-level logic ROS node, tailored to the needs of the current subtask to be automated. The iRob Surgical Automation Framework is available at `https://github.com/ABC-iRobotics/irob-saf`, and is being continuously developed and updated.

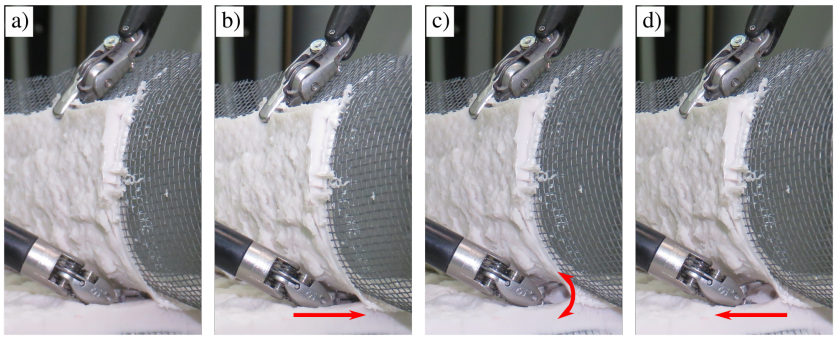

Figure 8
Motion primitives of the surgical subtask automation. a) The surgical instrument (large needle driver) moves to the dissection target; b) the robot pushes the instrument into the phantom; c) the instrument is opened; d) the robot pulls out the instrument.
Image credit: [17].

The framework can help in the implementation of further, more complex subtasks. In such development, it is straightforward to add new, necessary surgemes—like clipping or suturing. The implementation of new subtask can be added to the `irob_motion` package easily. Based on the our experiences, the most challenging aspect in automating more complex subtasks is the perception estimation of the environment, as computer vision usually struggles with light reflections or moving, deformable and hardly recognizable tissue, even in phantom environment, or ex vivo.

Acknowledgement

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. Partial support of this work comes from the Hungarian State and the European Union under the EFOP-3.6.1-16-2016-00010 project. T. D. Nagy and T. Haidegger are supported through the New National Excellence Program of the Ministry of Human Capacities. T. Haidegger is a Bolyai Fellow of the Hungarian Academy of Sciences.

References

[1] Á. Takács, D. Á. Nagy, I. J. Rudas, and T. Haidegger. Origins of Surgical Robotics: From Space to the Operating Room. *Acta Polytechnica Hungarica*, 13(1):13–30, 2016.

[2] 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 Intl. Conf. on Robotics and Automation*, pages 6434–6439, Hong Kong, 2014.

[3] P. Kazanzides. Open Source Software Libraries for Computer Integrated Surgery. https://cisst.org/wp-content/uploads/2016/03/YR_8_Open_Source_Software.pdf, 2005.

[4] Z. Chen, A. Deguet, R. H. Taylor, and P. Kazanzides. Software Architecture of the Da Vinci Research Kit. In *Proc. of the IEEE Intl. Conf. on Robotic Computing (IRC)*, pages 180–187, Taichung City, Taiwan, 2017.

[5] T. Haidegger. Autonomy for Surgical Robots: Concepts and Paradigms. *IEEE Trans. on Medical Robotics and Bionics*, 1(2):65–76, 2019.

[6] D. Á. Nagy, I. J. Rudas, and T. Haidegger. OntoFlow, a software tool for surgical workflow recording. In *2018 IEEE 16th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pages 119–124, Kosice, 2018.

[7] A. Murali, S. Sen, B. Kehoe, A. Garg, S. McFarland, S. Patil, W. D. Boyd, S. Lim, P. Abbeel, and K. Goldberg. Learning by observation for surgical subtasks: Multilateral cutting of 3D viscoelastic and 2D Orthotropic Tissue Phantoms. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation*, pages 1202–1209, Seattle, 2015.

[8] S. Sen, A. Garg, D. V. Gealy, S. McKinley, Y. Jen, and K. Goldberg. Automating multi-throw multilateral surgical suturing with a mechanical needle guide and sequential convex optimization. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation*, pages 4178–4185, Stockholm, 2016.

[9] Hyosig Kang and J. T. Wen. EndoBot: A robotic assistant in minimally invasive surgeries. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA 2001)*, volume 2, pages 2031–2036, Seoul, South Korea, May 2001.

[10] A. Garg, S. Sen, R. Kapadia, Y. Jen, S. McKinley, L. Miller, and K. Goldberg. Tumor localization using automated palpation with Gaussian Process Adaptive Sampling. In *Proc. of the 2016 IEEE Intl. Conf. on Automation Science and Engineering (CASE)*, pages 194–200, Fort Worth, 2016.

[11] K. A. Nichols and A. M. Okamura. Autonomous robotic palpation: Machine learning techniques to identify hard inclusions in soft tissues. In *Proc. of the 2013 IEEE Intl. Conf. on Robotics and Automation*, pages 4384–4389, Karlsruhe, 2013.

[12] S. McKinley, A. Garg, S. Sen, D. V. Gealy, J. McKinley, Y. Jen, and K. Goldberg. Autonomous Multilateral Surgical Tumor Resection with Interchangeable Instrument Mounts and Fluid Injection Device. 2016.

[13] B. Kehoe, G. Kahn, J. Mahler, J. Kim, A. Lee, A. Lee, K. Nakagawa, S. Patil, W. D. Boyd, P. Abbeel, and K. Goldberg. Autonomous multilateral debridement with the Raven surgical robot. In *Proc. of the 2014 IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 1432–1439, Hong Kong, 2014.

[14] D. Seita, S. Krishnan, R. Fox, S. McKinley, J. Canny, and K. Goldberg. Fast and Reliable Autonomous Surgical Debridement with Cable-Driven Robots Using a Two-Phase Calibration Procedure. In *Proc. of the 2018 IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 6651–6658, Brisbane, 2017.

[15] F. Richter, R. K. Orosco, and M. C. Yip. Open-Sourced Reinforcement Learning Environments for Surgical Robotics. *arXiv preprint arXiv:1903.02090.*, March 2019.

[16] A. Shademan, R. S. Decker, J. Opfermann, S. Leonard, A. Krieger, and P. C. W. Kim. Supervised autonomous robotic soft tissue surgery. *Science Translational Medicine*, 8(337), 2016.

[17] R. Elek, T. D. Nagy, D. Á. Nagy, T. Garamvölgyi, B. Takács, P. Galambos, J. K. Tar, I. J. Rudas, and T. Haidegger. Towards surgical subtask automation—blunt dissection. In IEEE, editor, *Proc. of the IEEE 21st Intl. Conf. on Intelligent Engineering Systems*, pages 253–258, Larnaca, 2017.

[18] D. Á. Nagy, T. D. Nagy, R. Elek, I. J. Rudas, and T. Haidegger. Ontology-Based Surgical Subtask Automation, Automating Blunt Dissection. *Journal of Medical Robotics Research*, 3(3), 2018.

[19] T. D. Nagy, M. Takács, I. J. Rudas, and T. Haidegger. Surgical Subtask Automation—Soft Tissue Retraction. In *Proc. of the 16th IEEE World Symposium on Applied Machine Intelligence and Informatics*, pages 55–60, Kosice, 2018.

[20] T. D. Nagy and T. Haidegger. An Open-Source Framework for Surgical Subtask Automation. In *Proc. of the ICRA 2018 IEEE Intl. Conf. on Robotics and Automation, Workshop on Supervised Autonomy in Surgical Robotics*, Brisbane, 2018.

[21] R. Sznitman, C. Becker, and P. Fua. Fast Part-Based Classification for Instrument Detection in Minimally Invasive Surgery. In P. Golland, N. Hata, C. Barillot, J. Hornegger, and R. Howe, editors, *Proc. of the Medical Image Computing and Computer-Assisted Intervention – MICCAI 2014*, volume 8674, pages 692–699, Cham, 2014. Springer Intl. Publishing.

[22] M. Allan, P.-L. Chang, S. Ourselin, D. J. Hawkes, A. Sridhar, J. Kelly, and D. Stoyanov. Image Based Surgical Instrument Pose Estimation with Multi-class Labelling and Optical Flow. In N. Navab, J. Hornegger, W. M. Wells, and A. Frangi, editors, *Proc. of the Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, volume 9349, pages 331–338, Cham, 2015. Springer Intl. Publishing.

[23] Yuan-Fang Wang, D. R. Uecker, and Wang Yulun. Choreographed scope manoeuvring in robotically-assisted laparoscopy with active vision guidance. In *Proc. of the Third IEEE Workshop on Applications of Computer Vision. WACV'96*, pages 187–192, Sarasota, US, December 1996.

[24] A. Krupa, J. Gangloff, C. Doignon, M. F. de Mathelin, G. Morel, J. Leroy, L. Soler, and J. Marescaux. Autonomous 3-D positioning of surgical instruments in robotized laparoscopic surgery using visual servoing. *IEEE Transactions on Robotics and Automation*, 19(5):842–853, October 2003.

[25] D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, D. Stoyanov, M. V. Scarzanella, P. Pratt, and G.-Z. Yang. Real-Time Stereo Reconstruction in Robotically Assisted Minimally Invasive Surgery. In T. Jiang, N. Navab, J. P. W. Pluim, and M. A. Viergever, editors, *Proc. of the Medical Image*

Computing and Computer-Assisted Intervention – MICCAI 2010, volume 6361, pages 275–282, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[26] J. J. Abbott, P. Marayong, and A. M. Okamura. Haptic Virtual Fixtures for Robot-Assisted Manipulation. In S. Thrun, R. Brooks, and H. Durrant-Whyte, editors, *Robotics Research*, volume 28, pages 49–64. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[27] M. Selvaggio, G. A. Fontanelli, F. Ficuciello, L. Villani, and B. Siciliano. Passive Virtual Fixtures Adaptation in Minimally Invasive Robotic Surgery. *IEEE Robotics and Automation Letters*, 3(4):3129–3136, October 2018.

[28] N. Abolhassani, R. Patel, and M. Moallem. Needle insertion into soft tissue: A survey. *Medical Engineering & Physics*, 29(4):413–431, 2007.

[29] S. S. Vedula, A. O. Malpani, L. Tao, G. Chen, Y. Gao, P. Poddar, N. Ahmidi, C. Paxton, R. Vidal, S. Khudanpur, G. D. Hager, and C. C. Chen. Analysis of the Structure of Surgical Activity for a Suturing and Knot-Tying Task. *PLOS ONE*, 11(3):e0149174, 2016.

[30] L. MacKenzie, J. A. Ibbotson, C. G. L. Cao, and A. J. Lomax. Hierarchical decomposition of laparoscopic surgery: A human factors approach to investigating the operating room environment. *Minimally Invasive Therapy & Allied Technologies*, 10(3):121–127, 2001.

[31] Y. Gao, S. S. Vedula, C. E. Reiley, N. Ahmidi, B. Varadarajan, H. C. Lin, L. Tao, L. Zappella, B. Béjar, and D. D. Yuh. JHU-ISI gesture and skill assessment working set (JIGSAWS): A surgical activity dataset for human motion modeling. In *Proc. of the MICCAI Workshop: M2CAI*, volume 3, Boston, 2014.

[32] B. Hannaford, R. Bly, I. Humphreys, and M. Whipple. Behavior Trees as a Representation for Medical Procedures. *arXiv:1808.08954 [cs]*, August 2018.

[33] ROS package for camera calibration. http://wiki.ros.org/camera_calibration.

[34] ROS package for stereo image processing. http://wiki.ros.org/stereo_image_proc.

[35] 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*, 47(6):2280–2292, 2014.

[36] K. Strobl and G. Hirzinger. Optimal Hand-Eye Calibration. In *Proc. of the 2006 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pages 4647–4653, Beijing, China, October 2006. IEEE.

[37] B. Bellekens, V. Spruyt, R. Berkvens, and M. Weyn. A Survey of Rigid 3D Pointcloud Registration Algorithms. pages 8–13, 2014.

[38] N. Ho. Optimal Rigid/Euclidean transform in 3D space. http://nghiaho.com/uploads/code/rigid_transform_3D.m.

[39] GStreamer: Open source multimedia framework. https://gstreamer.freedesktop.org/.