A Realistic Surgical Simulator for Non-Rigid and Contact-Rich Manipulation in Surgeries with the da Vinci Research Kit

Yafei Ou, *Student Member, IEEE*, Sadra Zargarzadeh*, *Student Member, IEEE*, Paniz Sedighi*, *Student Member, IEEE*, and Mahdi Tavakoli, *Senior Member, IEEE*

Abstract— Realistic real-time surgical simulators play an increasingly important role in surgical robotics research, such as surgical robot learning and automation, and surgical skills assessment. Although there are a number of existing surgical simulators for research, they generally lack the ability to simulate the diverse types of objects and contact-rich manipulation tasks typically present in surgeries, such as tissue cutting and blood suction. In this work, we introduce CRESSim, a realistic surgical simulator based on PhysX 5 for the da Vinci Research Kit (dVRK) that enables simulating various contact-rich surgical tasks involving different surgical instruments, soft tissue, and body fluids. The real-world dVRK console and the master tool manipulator (MTM) robots are incorporated into the system to allow for teleoperation through virtual reality (VR). To showcase the advantages and potentials of the simulator, we present three examples of surgical tasks, including tissue grasping and deformation, blood suction, and tissue cutting. These tasks are performed using the simulated surgical instruments, including the large needle driver, suction irrigator, and curved scissor, through VR-based teleoperation.

I. INTRODUCTION

High-performance and realistic real-time simulators are playing an increasingly significant role in robotics research, not only because they serve as a playground for testing control and automation algorithms without the need for real robots, but also because a large number of recent advances using machine learning (ML) approaches are heavily driven by simulators that provide synthetic data.

However, developing real-time simulators for surgical robots poses unique challenges compared with other general robotics scenarios, due to the nature of surgeries that require a diverse and unique range of contact-rich manipulation. Unlike general robotics such as autonomous driving and home service robots, multiple types of objects, including rigid bodies (e.g. surgical instruments), soft bodies (e.g. voluminous soft tissue), fluids (e.g. blood and other body fluids), and cloth-type tissue (e.g. fascia) are commonly

Yafei Ou, Sadra Zargarzadeh, and Paniz Sedighi are with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, Alberta, Canada. {yafei.ou, sadra.zar, sedighi1}@ualberta.ca

Mahdi Tavakoli is with the Department of Electrical and Computer Engineering and the Department of Biomedical Engineering, University of Alberta, Edmonton, Alberta, Canada. mahdi.tavakoli@ualberta.ca

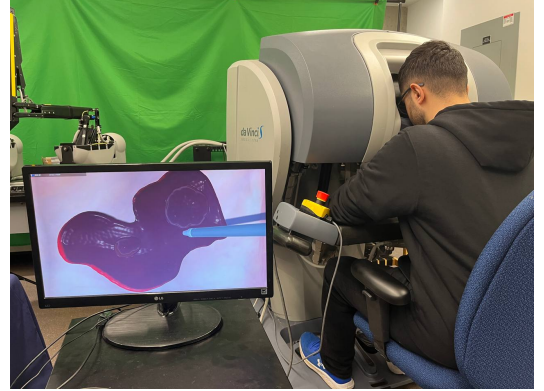*Sadra Zargarzadeh and Paniz Sedighi contributed equally.

Fig. 1: Teleoperating the CRESSim environment using the dVRK console.

present in one single surgical scene, contacting with each other. Furthermore, complex manipulations, such as cutting, cauterization, and fluid suctioning do not usually exist in other general robotics simulation scenes. These factors lead to the need for specific adaptations and designs for surgical simulators. For instance, to simulate cauterization, burning and smoking effects must be included.

With the help of the da Vinci Research Kit (dVRK) and its open-sourced software [1], surgical robotics research has gained increasing attention. While there are a number of existing surgical simulators featuring the da Vinci system, they are either only for commercial surgical training purposes or limited to specific scenarios and cannot simulate various types of objects presented typically in a surgery. The da Vinci SimNow[1] is a commercial surgical skills training simulator developed by Intuitive Surgical, Inc. (Sunnyvale, California). Although it supports a variety of training tasks such as vessel clipping and tissue cutting, it is only for training surgeons using proprietary robotic platforms and does not provide an open interface for robotics research. Asynchronous Multi-Body Framework (AMBF) [2] is an open-sourced simulator based on Bullet Physics [3], which supports rigid and soft body simulation through finite element method (FEM). However, while Bullet provides high-performance rigid-body simulation, the FEM soft body is less realistic. Furthermore, it is unable to simulate fluids, which usually exist in surgical environments such as blood and other body fluids. Similarly, Assisted Teleoperation with Augmented Reality (ATAR) [4]

---

[1]https://www.intuitive.com/products-and-services/da-vinci/learning/simnow/

and SurRoL [5] are also based on Bullet, which have the same limitations.

Some other surgical simulators are developed based on the Simulation Open Framework Architecture[2] (SOFA), such as LapGym [6]. However, SOFA sacrifices computation costs for a more accurate FEM performance than Bullet, making simulating large-scale and complex surgical scenes slow. Furthermore, it does not support fluids as well. UnityFlexML, on the other hand, utilizes Nvidia FleX for Unity, a position-based dynamics (PBD) engine plugin for Unity, to simulate soft tissue manipulation [7]. While diverse objects including rigid bodies, soft bodies, cloths, and fluids are natively supported by the FleX engine, the discontinuation of FleX for Unity makes it inadvisable and inconvenient to develop new environments and new manipulations such as cutting. Furthermore, PBD soft bodies are less realistic compared with FEM soft bodies, especially when large position displacements or velocities are applied to the particles.

In this work, we present **CRESSim**, a **C**ontact-**R**ich **E**nvironment for **S**urgical **Sim**ulation. CRESSim is a realistic simulator that enables the simulation of various contact-rich manipulation tasks in surgeries. The developed system is built on Unity and PhysX 5 SDK, allowing the simulation of rigid bodies, serial robots, soft bodies, cloth, fluid, and complex manipulation tasks such as cutting. The main contributions are as follows:

- We build a new platform for surgical simulation featuring the dVRK using Unity and PhysX 5 SDK. A Unity native plugin is developed to enable GUI-based interactive editing of simulation scenes, allowing adding and removing various objects and constraints.
- We introduce two new simulated surgical instruments and three contact-rich surgical tasks in the proposed simulator. These include tissue grasping and manipulation, blood suction (single-arm tasks), and tissue cutting (a bimanual task).
- We incorporate the real-world dVRK console to allow teleoperating the simulated robots through virtual reality (VR), and present preliminary demonstrations for completing the three contact-rich tasks using teleoperation.

To the best of the authors' knowledge, this is also the first time that the PhysX 5 library has been integrated into Unity for building a surgical simulation platform.

II. RELATED WORK

A. Surgery and Surgical Robot Simulation

Existing surgical simulators can be categorized into commercial ones such as the da Vinci SimNow and VirtaMed LaparoS[3] which are mainly used for training surgeons, and open-source research platforms such as LapGym [6], AMBF [2], ATAR [4], and V-Rep Simulator for the dVRK [8], that focus on providing a multi-faceted interface for robotics research. Other surgical simulators such as SurRoL [5], dVRL
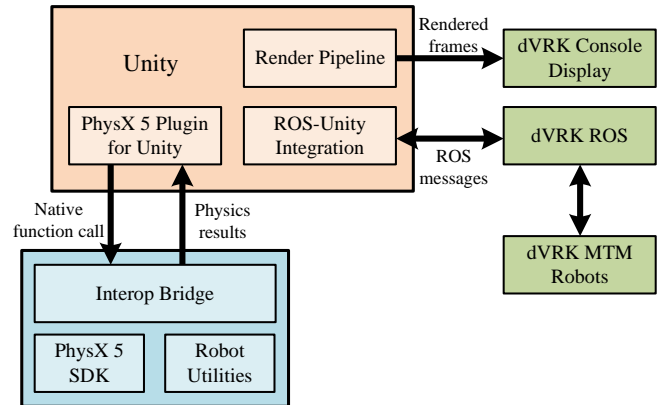
Fig. 2: System architecture.

[9], AMBF-RL [10], and UnityFlexML [7] have specific focuses on machine learning applications. Table I shows a comparison between the proposed simulation environment and the existing surgical simulators.

B. Simulation for Non-rigid and Contact-Rich Manipulation

There are a number of simulators for non-rigid and contact-rich manipulation in the general robotics field. However, most of them are specific to a particular type of manipulation, such as fluid manipulation. To simulate fluids, [11] incorporates the material point method (MPM) into SAPIEN [12], a PhysX 4-based simulator that only supports rigid body dynamics. FluidLab [13] proposes FluidEngine based on MPM that covers a wider range of fluids. SoftGym [14] builds on top of Nvidia FleX, and TDW [15] integrates PhysX 4 with Nvidia FleX to support soft bodies, cloth, and fluids. However, due to the discontinuation of FleX and Flex for Unity, it is inconvenient to develop new simulators based on it, especially when developing customized manipulation behavior such as cutting. DiffSim [16] specializes in soft-body deformation, such as cloth folding. DiSECt [17] has a specific focus on cutting voluminous soft materials using FEM.

The advancement of simulators has contributed to the development of methods for the automatic control of robots in completing contact-rich daily tasks, such as pouring [18], cloth folding [19], [20], and soft object deformation [21]. Although these studies represent progress in simulation environments and robot automation in the general robotics field, they are specific to a limited range of objects and manipulation types. However, surgical scenes require the presence of various types of objects and the simulation of different manipulation tasks (grasping, cutting, fluid suction, burning, and more) in a single scene, making it impractical to directly use these simulators for surgical simulation.

III. SYSTEM ARCHITECTURE

A. Overview

The developed surgical simulator is based on Nvidia PhysX 5 SDK, the latest version of the Nvidia PhysX physics engine. Compared with previous versions that only support

TABLE I: Comparison of CRESSim with existing surgical simulators

Platform	Physics[1]	Manipulation[2]	Robot integration[3]
AMBF, ATAR (Bullet)	R, S_F	R, S	D^+, L, T
SurRoL, V-Rep Simulator for the dVRK (Bullet)	R, S_F^*	R, S^*	D^+, L, T
LapGym (SOFA)	R, S_F, C_F	R, S, C, C^+	G
UnityFlexML (FleX)	R_P, S_P, C_P, F_P	R, S, C^*, C^{+*} F^*	D, L, T
CRESSim (PhysX 5)	R, R_P^*, S_F, S_P^*, C_F^*, C_P, F_P	R, S, C, C^+, F	D^+, L, C, S, T

[1] Physics: regular rigid body (R), PBD rigid body (R_P), FEM soft body (S_F), PBD soft body (S_P), FEM cloth (C_F), PBD cloth (C_P), and position-based fluids (F_P).
[2] Manipulation: rigid object grasping (R), soft object grasping and deformation (S), cloth grasping and deformation (C), cloth cutting (C^+), and fluid manipulation (F).
[3] Robot integration: kinematic-only dVRK robots (D), dVRK robots with dynamics (D^+), large needle driver (L), curved scissor (C), suction irrigator (S), general laparoscopic tools (G), and teleoperation (T).
* Items marked with ∗: it is technically possible but has not been implemented or used in the platform.

rigid body dynamics, PhysX 5 now supports a wide variety of physical simulations with GPU optimization, including (a) rigid bodies, (b) soft bodies using FEM, and (c) cloth, inflatables, and fluids using PBD. It also allows the simulation of serial robots using articulation joints. With PhysX 5 as the low-level physics engine, our simulation environment is built in Unity, a 3D game development software that has been widely used in surgical simulations. A Unity native plugin is developed to incorporate the PhysX engine. It is worth noting that the current built-in 3D physics engine of Unity is PhysX 4.1, which does not support FEM soft bodies and PBD-based objects. With the native support of PhysX 5 across platforms, we achieve cross-platform compatibility on both Windows and Linux.

Fig. 2 shows an overview of the system architecture. *PhysX 5 Plugin for Unity* is a Unity native plugin that and communicates with a native library *Interop Bridge* to set the physics scene configuration and obtain the physics results. The plugin implements a number of custom script components for the Unity Inspector window to allow for GUI-based scene editing, as shown in Fig. 3. Unity's *ROS-Unity Integration* package is used to achieve ROS communication with the *dVRK-ROS* topics to interact with the physical master tool manipulator (MTM) robot from the dVRK. Using the *Render Pipeline* provided by Unity, we can obtain stereo-vision frames and display them on the dVRK console displays.
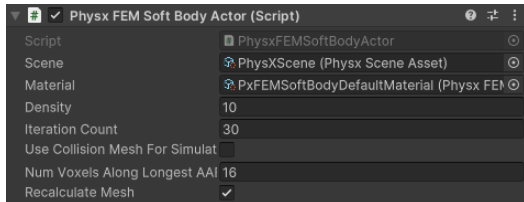
Fig. 3: Inspector window for defining an FEM soft body.

B. Physics Functionalities

We utilize the physics functionalities provided by PhysX 5 to simulate a large variety of objects, including rigid bodies, serial robots, soft bodies, cloth, and fluid. To avoid undesir-

able behavior, Unity's built-in physics simulation which uses PhysX 4.1 is disabled.
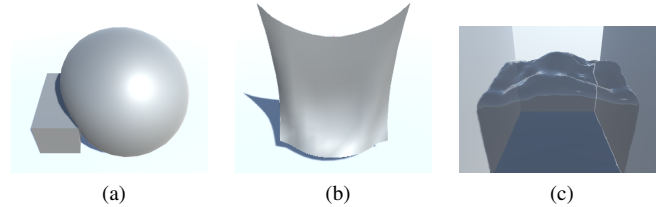
Fig. 4: Examples of simulated objects. (a) Rigid and soft bodies; (b) Cloth with fixed vertices; (c) Fluid.

1) Rigid body and robot: PhysX provides a good foundation for rigid body dynamics. Taking advantage of this, CRESSim supports three types of rigid objects: static, dynamic, and kinematic rigid bodies. Furthermore, by utilizing the articulation joints, a type of joint that enables zero joint error using reduced coordinates, serial robots are supported by connecting rigid links through the articulation joints. Each articulation joint is driven by a PD controller. Common useful functions are implemented for serial robots, including forward kinematics, spatial and body Jacobians, and Jacobian-based numerical inverse kinematics. These are part of the *Robot Utilities* component.

2) FEM Soft body: Leveraging GPU-based FEM soft body simulation provided by PhysX, we are able to simulate soft bodies such as tissue in surgical scenes. Parameters related to the physics properties of the soft body can be controlled, including Young's modulus, Poisson's ratio, and friction. Fig. 4a shows a soft sphere colliding with a rigid box.

3) PBD Cloth and Fluid: The PBD particle system is also supported by PhysX with GPU. PBD is an efficient approach for simulating large amounts of particle system-based objects, including cloth and fluids, as shown in Fig. 4b and 4c. In our implementation, both cloth and fluids include a set of 12 parameters for modifying the PBD material, such as friction, damping, and adhesion. In the case of fluid, buoyancy is an additional parameter that needs to be set. When using PBD cloth, each particle is connected with

neighboring ones with constraints such as damped springs. Direct modification to the positions and velocities of the particles, as well as the spring constraints, are also supported during the simulation.

4) Manipulation and Contact Features: Through direct access to the low-level PhysX library functions during simulation, a number of manipulation and customized contacts can be achieved. Using rigid-soft and rigid-particle attachments, we can achieve soft-body grasping and cloth grasping, which is common in surgical scenes. Furthermore, by removing the particle spring constraints in real time, cloth cutting can be efficiently simulated. Customized force fields that are applied to the particles are also implemented by modifying the particle positions and velocities at each simulation step.

C. Teleoperation in Virtual Reality

The real dVRK console display and the MTM robots are incorporated into the simulator to allow for teleoperating the simulated robots through VR. To allow communication between the simulator and the real MTM robots, *ROS-Unity Integration* is used and the dVRK-specific ROS messages are generated in Unity. The process for single-arm teleoperation is summarized in Algorithm 1, where `AlignMTMWithPSM` is a Unity coroutine that waits until the MTM is positioned in the desired pose with the same orientation as the simulated patient side manipulator (PSM) robot. Each joint of the PSM robot is controlled by a PD controller. Since Unity's `FixedUpdate` function is used which is called every 0.02 seconds, the teleoperation is performed at a frequency of 50 Hz. During teleoperation, two cameras with a distance along the horizontal axis are present in the simulation scene to render two frames for stereo vision. The frames are shown on the two displays in the dVRK console, providing a VR experience. Fig. 1 shows the teleoperation of a simulated PSM in the blood suction scene by a human operator.

IV. EXAMPLE SURGICAL SCENES

A. Simulated PSM with Various Surgical Instruments

In this work, the PSM from the dVRK is simulated. The 3D models are modified from those in [5] with improvements on mesh normals for rendering. As discussed in Section III-B.1, the PSM robot is defined by a series of rigid links through the reduced coordinate joints, with masses and inertia calculated from their bounding convex hull. Using the functionalities implemented in the *Robot Utilities*, we are able to achieve both joint and Cartesian space control of the robot. The simulated PSM with the large needle driver is shown in Fig. 5a.

Besides the large needle driver that has been simulated in various existing dVRK simulators, we additionally simulate two more surgical instruments: the curved scissor and the suction irrigator tool, as shown in Fig. 5b to 5d. These tools are used for simulating the complex contact-rich surgical tasks discussed in Section IV-B.

Algorithm 1 Teleoperation using real MTMs

```
initializePSM ← true
initializeMTM ← true
isTeleoperating ← false
for each FixedUpdate call do
    if initializePSM then
        Initialize PSM
        initializePSM ← false
    end if
    if initializeMTM and current time ≥ delay then
        Start AlignMTMWithPSM coroutine
        initializeMTM ← false
        if AlignMTMWithPSM ends then
            isTeleoperating ← true
        end if
    end if
    if isTeleoperating then
        P_PSM ← P_PSM + scale × ΔP_MTM
        R_PSM ← R_MTM
        J ← InverseKinematics(P_PSM, R_PSM)
        DriveJoints(J)
    end if
end for
```

B. Simulated Contact-Rich Surgical Tasks

1) Soft Tissue grasping and manipulation: The task focuses on grasping and deforming soft tissue, a common sub-task in surgeries. We emulate the scene of a hysterectomy, where the uterus, ovary, and fat tissue are present, as shown in Fig. 6a. The goal of this task is to grasp and deform the organs to relocate some parts of the tissue for other sub-tasks such as cauterization, or to reveal the area underneath the tissue. The organs and tissue present in the scene are simulated as FEM soft bodies, and grasping is achieved by attaching a mesh vertex to the grasping tool. The task is inspired by [7], as well as a hysterectomy simulation scene in VirtaMed LaproS.

2) Blood suction: Blood suction is another common surgical sub-task where the goal is to use the suction irrigator tool to remove the blood in the scene, as shown in Fig. 7a. The background tissue is a large FEM soft body, and the blood is simulated by the PBD fluid. Suction is achieved by applying a customized force field around the suction tooltip. PBD particles are removed once they are close enough to the tooltip. This task is inspired by [22].

3) Tissue cutting: In this task, a soft tissue with a thin layer of fascia is present and the goal is to grasp and pull the fascia and cut it along a given line, as shown in Fig. 8a. This task is bimanual and is commonly present in surgeries, although the cutting procedure may be replaced by cauterization. The background tissue is an FEM soft body. The spherical tissue for grasping and cutting consists of two parts: an FEM soft body inside which emulates the main part of the organ, and a thin layer of PBD cloth surrounding the organ, which simulates the fascia. The cloth layer has
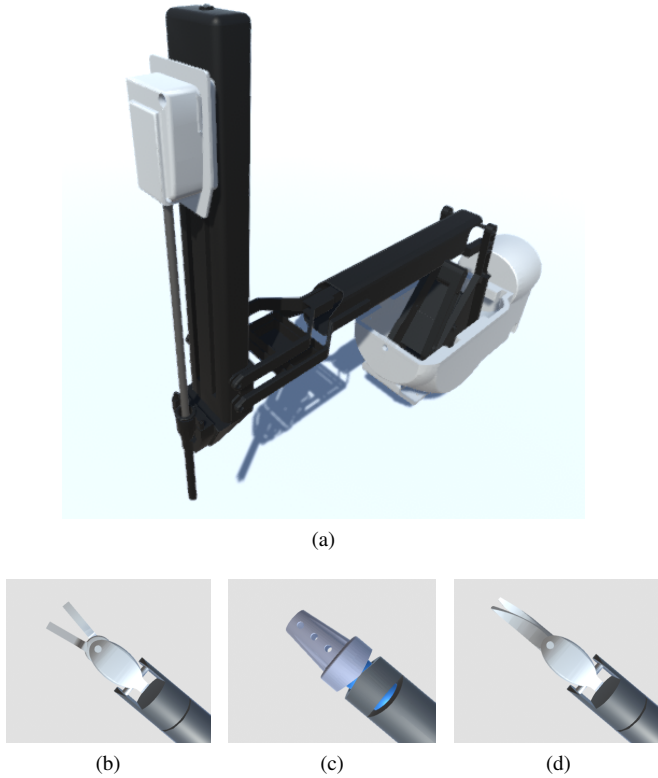
(a)

(b)   (c)   (d)

Fig. 5: Simulated PSM. (a) PSM robot; (b) Large needle driver; (c) Suction irrigator; (d) Curved scissor.

stretched spring constraints between particles to emulate the elastic behavior of a fascia. To simulate the cutting procedure, Algorithm 2 is used to update the particle springs at each `FixedUpdate` function call, and the triangle mesh for rendering the tissue is updated accordingly. In practice, additional steps are needed to address issues related to triangle mesh welding. Furthermore, parallelism is needed for efficiently modifying the mesh in real time, due to the large number of particles and triangles needed for processing. The design of this task is inspired by similar tasks in da Vinci SimNow.

Algorithm 2 Fascia tissue cutting

---

for each `FixedUpdate` call **do**
    for each particle spring constraint s **do**
        $\mathbf{p}_0 \leftarrow$ first particle position
        $\mathbf{p}_1 \leftarrow$ second particle position
        if $\overrightarrow{\mathbf{p}_0\mathbf{p}_1}$ intersects with cutting curve **then**
            remove spring s
        end if
    end for
    for each render mesh triangles t **do**
        if no springs exist in at least two edges **then**
            remove triangle t
        end if
    end for
end for

To evaluate the quality of the designed simulation scenes, we perform the tasks using teleoperation and record the videos[4]. Please note that the scenes are designed to showcase the capability of the simulator in terms of the wide range of simulation objects, the achievable manipulation types, and the render capabilities. These scenes are not intended to be as accurate as possible to any specific surgical scenes, nor that they mimic the body of an actual patient, as designing actual surgical scenes requires extra 3D modeling of the tissue and organs, as well as art design of the materials and textures, which are out of the scope of this research.

Fig. 6 to 8 show the screenshots taken from the teleoperation trials for three tasks. As shown in the figures, in all three tasks, we experienced close-to-real interactions in terms of the contact and collisions between different objects (the robots and the simulated tissue and organs) as well as the manipulation behavior. In soft tissue grasping and deformation, the tissue deforms in response to the motion of the robot grasper. In blood suction, blood is suctioned towards the suction tool and correctly removed from the scene. In tissue cutting, we are able to perform bimanual grasping and cutting on the tissue by grasping and stretching a point on the fascia using the left arm and cutting it along the line using the right arm. The contact and collisions between simulated objects significantly enhance the realism of the scenes, especially the collision between two robot arms, the collision between the robot arm and the soft tissue and organs, and the contact between soft tissue and organs.

V. Discussion

Although the physics computation for each `FixedUpdate` can take longer time when FEM and PBD objects are present, preliminary profiling results show that the simulator can run at least 50 to 60 frames per second (FPS), and we did not experience noticeable FPS drops during the teleoperation testing. Table II shows the time spent on physics simulation at each step and the post-processing time for meshes and particles. Samples are taken from 10 consecutive physics updates, in which all `FixedUpdate` functions in scripts present in the scene are called. Tests are conducted on a PC with Intel Core i7-12700F and Nvidia RTX 3070. As expected, blood suction and tissue cutting scenes add more physics computation time due to the added complexity of the scene with both FEM and PBD objects, and the additional time needed for processing the meshes and particles. However, as the code is not fully optimized for computational efficiency, there is room for improvement in future work, for example, by adding more parallelism. We have also noticed slight delays in teleoperation, primarily because the PD gains for the robot joint controllers are not well-tuned to rapidly drive the joint angles to the setpoint. This will be addressed in future work as well.

With Unity as an intuitive platform for defining the simulated scenes through GUI, it is straightforward to further

---

[4]https://drive.google.com/drive/folders/1oHEhEt4K9kQYQMF5rYfPA6bthrXBM7gg
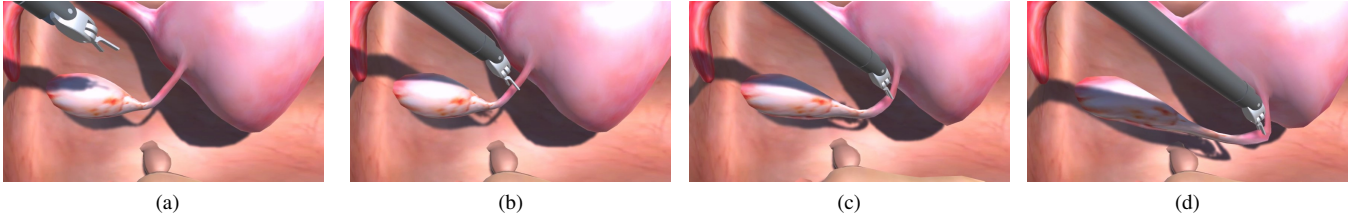
Fig. 6: (a) to (d) is a sequence of screenshots taken from the tissue grasping and deformation task during teleoperation.
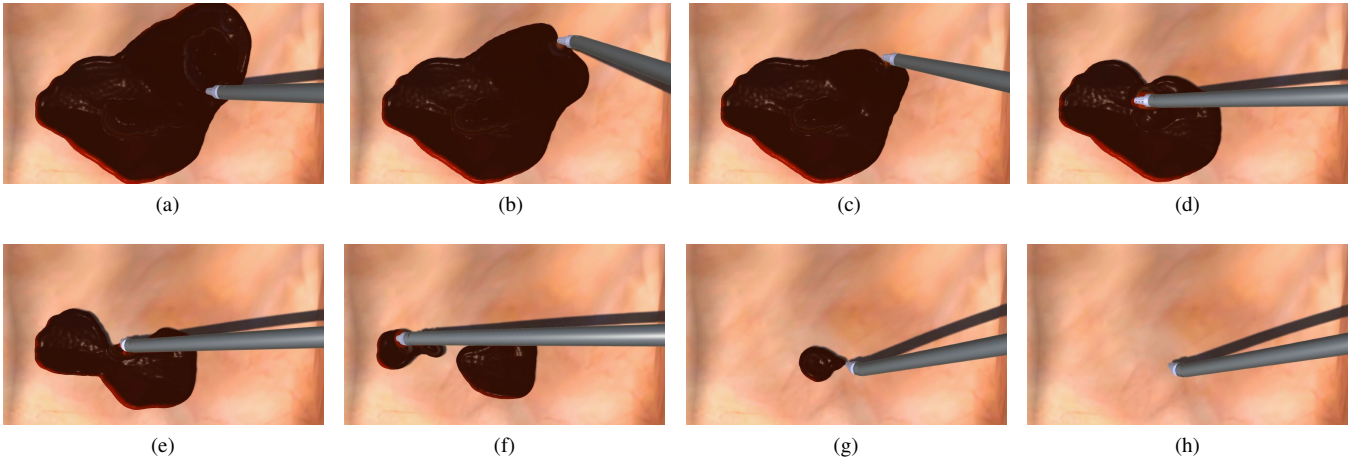

Fig. 7: (a) to (h) is a sequence of screenshots taken from the blood suction task during teleoperation.
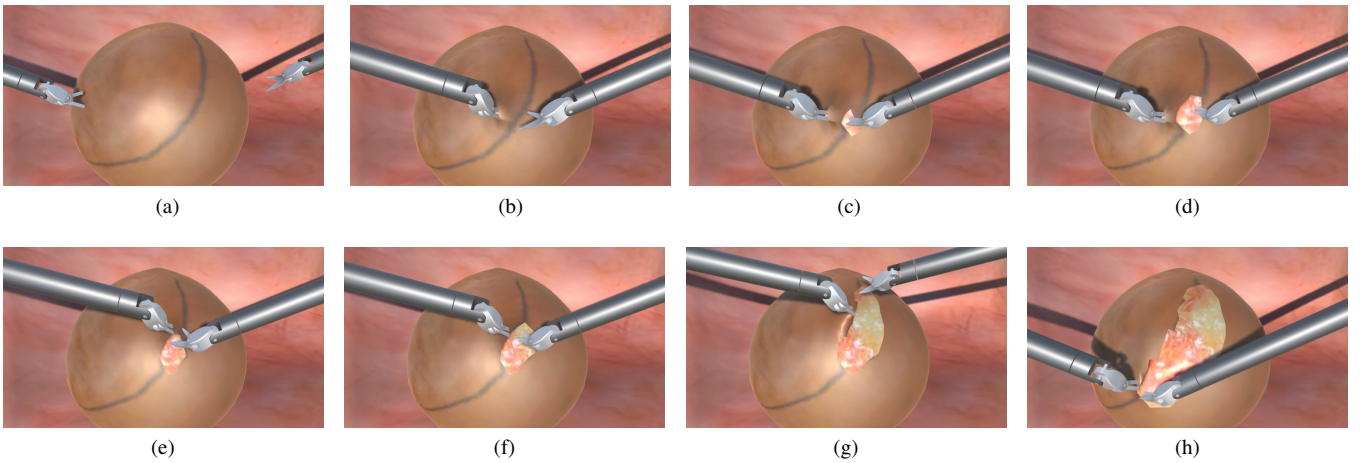

Fig. 8: (a) to (h) is a sequence of screenshots taken from the tissue cutting task during teleoperation.

TABLE II: Physics simulation and post-processing time.

Scenes	Physics advance (ms)	Mesh and particle post-processing (ms)	FixedUpdate total (ms)
Tissue grasping and deformation	9.51 ± 0.63	0.12 ± 0.02	9.68 ± 0.63
Blood suction	14.00 ± 0.31	0.19 ± 0.03	14.24 ± 0.31
Tissue cutting	15.86 ± 0.39	0.39 ± 0.16	16.52 ± 0.81

create realistic scenes for real surgical tasks, as long as the tissue and surgical instruments are represented in 3D models. The long-term objective of CRESSim is to emulate the functionalities provided by the da Vinci SimNow and VirtaMed LaproS to allow for simulating various realistic surgical tasks, but with a particular focus on surgical robotics research instead of commercial usage. The source code will be made available to the community to facilitate research

in surgical automation and autonomy, as well as other applications that require realistic surgical task simulations. One specific application of the simulator is simulation-to-reality surgical robot learning using reinforcement learning and imitation learning algorithms.

While this work presents an initial implementation and three simulated scenes, we have not been able to conduct user studies on the quality and realism of the simulator. Future studies could include evaluations and feedback from surgeons, especially from users of the da Vinci Surgical System and the da Vinci SimNow. Additionally, other limitations need to be addressed in the future development of CRESSim. One major limitation is that there are a large number of surgical instruments and procedures, such as cauterization, that have not been covered in this work. Furthermore, although cutting can be simulated for the PBD cloth, which can be used to simulate cutting thin tissue such as fascia, FEM soft body cutting is not achieved. Simulating cutting for the FEM soft body requires extensive real-time re-calculation of the tetrahedron mesh, making it extremely challenging. This limits the simulation of cutting thick and voluminous soft tissue in surgeries, which is also a common procedure. We expect to address these limitations in the future work.

VI. Conclusion

In this work, we present CRESSim, a realistic surgical simulation environment for the dVRK that allows contact-rich manipulations. The simulator is built on Unity and PhysX 5 SDK, and the real dVRK console and MTMs are incorporated into the simulator for VR-based teleoperation. Thanks to the simulation capabilities provided by PhysX 5 compared with its previous versions, we are able to simulate soft bodies, cloth, inflatables, and fluids that exist in surgeries, and the contact-rich manipulation of these objects. To show the advantages and potentials of the developed simulator, we showcase 3 examples of surgical simulation scenes and their corresponding tasks, including tissue grasping and deformation, blood suction, and tissue cutting. Preliminary experiments and profiling show the platform's capability to simulate surgical tasks and allow real-time teleoperation. In future work, we will further enhance the simulator to cover more realistic surgical scenes and various surgical instruments.

References

[1] 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 *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 6434–6439.

[2] A. Munawar, Y. Wang, R. Gondokaryono, and G. S. Fischer, "A real-time dynamic simulator and an associated front-end representation format for simulating complex robots and environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 1875–1882.

[3] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016.

[4] N. Enayati, A. M. Okamura, A. Mariani, E. Pellegrini, M. M. Coad, G. Ferrigno, and E. De Momi, "Robotic assistance-as-needed for enhanced visuomotor learning in surgical robotics training: An experimental study," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6631–6636.

[5] J. Xu, B. Li, B. Lu, Y.-H. Liu, Q. Dou, and P.-A. Heng, "Surrol: An open-source reinforcement learning centered and dvrk compatible platform for surgical robot learning," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021.

[6] P. M. Scheikl, B. Gyenes, R. Younis, C. Haas, G. Neumann, M. Wagner, and F. Mathis-Ullrich, "Lapgym–an open source framework for reinforcement learning in robot-assisted laparoscopic surgery," *arXiv preprint arXiv:2302.09606*, 2023.

[7] E. Tagliabue, A. Pore, D. Dall'Alba, E. Magnabosco, M. Piccinelli, and P. Fiorini, "Soft tissue simulation environment to learn manipulation tasks in autonomous robotic surgery," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 3261–3266.

[8] G. A. Fontanelli, M. Selvaggio, M. Ferro, F. Ficuciello, M. Vendittelli, and B. Siciliano, "A v-rep simulator for the da vinci research kit robotic platform," in *2018 7th IEEE International Conference on Biomedical Robotics and Biomechatronics (Biorob)*. IEEE, 2018, pp. 1056–1061.

[9] F. Richter, R. K. Orosco, and M. C. Yip, "Open-sourced reinforcement learning environments for surgical robotics," *arXiv preprint arXiv:1903.02090*, 2019.

[10] V. M. Varier, D. K. Rajamani, F. Tavakkolmoghaddam, A. Munawar, and G. S. Fischer, "Ambf-rl: A real-time simulation based reinforcement learning toolkit for medical robotics," in *2022 International Symposium on Medical Robotics (ISMR)*. IEEE, 2022, pp. 1–8.

[11] J. Gu, F. Xiang, X. Li, Z. Ling, X. Liu, T. Mu, Y. Tang, S. Tao, X. Wei, Y. Yao, X. Yuan, P. Xie, Z. Huang, R. Chen, and H. Su, "Maniskill2: A unified benchmark for generalizable manipulation skills," in *International Conference on Learning Representations*, 2023.

[12] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang *et al.*, "Sapien: A simulated part-based interactive environment," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 097–11 107.

[13] Z. Xian, B. Zhu, Z. Xu, H.-Y. Tung, A. Torralba, K. Fragkiadaki, and C. Gan, "Fluidlab: A differentiable environment for benchmarking complex fluid manipulation," *arXiv preprint arXiv:2303.02346*, 2023.

[14] X. Lin, Y. Wang, J. Olkin, and D. Held, "Softgym: Benchmarking deep reinforcement learning for deformable object manipulation," in *Conference on Robot Learning*. PMLR, 2021, pp. 432–448.

[15] C. Gan, J. Schwartz, S. Alter, D. Mrowca, M. Schrimpf, J. Traer, J. De Freitas, J. Kubilius, A. Bhandwaldar, N. Haber *et al.*, "Threedworld: A platform for interactive multi-modal physical simulation," *arXiv preprint arXiv:2007.04954*, 2020.

[16] Y.-L. Qiao, J. Liang, V. Koltun, and M. C. Lin, "Scalable differentiable physics for learning and control," *arXiv preprint arXiv:2007.02168*, 2020.

[17] E. Heiden, M. Macklin, Y. Narang, D. Fox, A. Garg, and F. Ramos, "Disect: A differentiable simulation engine for autonomous robotic cutting," *arXiv preprint arXiv:2105.12244*, 2021.

[18] E. Babaians, T. Sharma, M. Karimi, S. Sharifzadeh, and E. Steinbach, "Pournet: Robust robotic pouring through curriculum and curiosity-based reinforcement learning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 9332–9339.

[19] T. Weng, S. M. Bajracharya, Y. Wang, K. Agrawal, and D. Held, "Fabricflownet: Bimanual cloth manipulation with a flow-based policy," in *Conference on Robot Learning*. PMLR, 2022, pp. 192–202.

[20] H. Ha and S. Song, "Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding," in *Conference on Robot Learning*. PMLR, 2022, pp. 24–33.

[21] F. Ficuciello, A. Migliozzi, E. Coevoet, A. Petit, and C. Duriez, "Fem-based deformation control for dexterous manipulation of 3d soft objects," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4007–4013.

[22] F. Richter, S. Shen, F. Liu, J. Huang, E. K. Funk, R. K. Orosco, and M. C. Yip, "Autonomous robotic suction to clear the surgical field for hemostasis using image-based blood flow detection," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1383–1390, 2021.