Using SciKit-Surgery for Augmented Reality in Surgery Research

Stephen Thompson, Miguel Xochicale, Thomas Dowrick and Matthew Clarkson

June 14, 2023

SciKit-Surgery for Augmented Reality Surgery Research

Stephen Thompson[1], Miguel Xochicale[1], Thomas Dowrick[1], and Matthew J. Clarkson[1]

[1]*Wellcome EPSRC Centre for Interventional and Surgical Sciences, University College London*
s.thompson@ucl.ac.uk

INTRODUCTION

SciKit-Surgery provides open source libraries to support research and translation of applications for augmented reality in surgery [1]. This paper discusses recent developments in SciKit-Surgery and case studies using SciKit-SurgeryBARD to support research into visualisation and user interface design for augmented reality in surgery [2], [3].

The availability of high quality software tools for research and translation is a key enabler for scientific progress. Research into surgical robotics, image guided surgery, and augmented reality for surgery brings together many disciplines and depends on a strong engineering base to provide the tools that researchers need (e.g., hardware interfaces, data management, data processing, visualisation, and user interfaces). SciKit-Surgery was conceived as a more accessible replacement for existing toolkits written predominantly in C++. Experience has taught us that whilst implementations in C++ could be robust and offer optimised performance, the need to learn the language and the difficulties of maintaining cross platform compilation presented a higher a barrier of entry for most researchers.

Whilst research software can be initially developed using short term research grants, the longer term sustainability of the software depends on other researchers being able to contribute to the software, both for maintenance and to introduce new features. For that to happen the software needs to be compact, written in a language that be easily interpreted by humans, and well documented. We conceived SciKit-Surgery to be a set of individual Python modules that could be used on their own by researchers to explore a specific topic or assembled into high quality applications that could be rapidly deployed to clinic to enable translation from bench to bedside.

MATERIALS AND METHODS

Python has grown to be the most used programming language for researchers[1]. Python combines an accessible syntax with a massive ecosystem of user contributed modules to provide ready made tools for most research tasks. Perhaps most important are the package management tools that enable modules to be installed and maintained on many platforms. For this reason we chose Python and as the base platform to develop and distribute the SciKit-Surgery libraries.

We made the key decision to keep our libraries highly modular. The intention is that each SciKit-Surgery library should do one thing and maintain orthogonality[4], so that modules could be combined into applications, without causing software conflicts. We developed a software template so that all SciKit-Surgery libraries share a familiar structure and continuous integration testing and deployment to PyPi is easilily managed. The SciKit-Surgery organisation on GitHub creates a space where all libraries are collated and discussion and contribution in enabled.

Easily Swappable Hardware: SciKit-SurgeryCore sits at the centre of the ecosystem, defining data and message types common to all the SciKit-Surgery libraries. A key part of our work is developing hardware interfaces to talk to clinical and tracking hardware as this is a common stumbling block for researchers. SciKit-SurgeryCore defines an abstract API for all clinical hardware so that one tracking or imaging system can be easily substituted with another. Figure 1 gives an example of this, swapping an NDI tracking system with the open source ArUco tracking system [5].

For robotic surgery, we have demonstrated how our standard tracking interface can be used to create a ROS publisher node. We are also intending to develop tracking classes to get tracking data from robot kinematics, rather than relying on third party trackers.

RESULTS

SciKit-Surgery has grown rapidly since 2020 and is being used by researchers in image guided interventions globally. We are also seeing a steadily growing pool of contributors via GitHub. There are currently over 40 libraries listed on the SciKit-Surgery home page. We have found that it is the hardware interface libraries (SciKit-SurgeryNDITracker and SciKit-SurgeryBK) that have attracted the most external interest.

Within our own research group the use of SciKit-Surgery has had a transformational effect on our more junior research students (BSc and MSc). By carefully scoping research projects we can get junior researchers to contribute code and research outputs in short (3 months) projects.

SciKit-SurgeryBARD: Our current research on computer human interaction is built using SciKit-SurgeryBARD, the Basic Augmented Reality Demonstrator. BARD links together tracking libraries with visualisation using SciKit-SurgeryVTK and a

[1]top-programming-languages-for-data-scientists-in-2022

python-based user interface (PySide) to enable students to quickly test their own augmented reality systems.

The majority of research into image guided surgery and augmented reality in surgery focuses on addressing the many technical challenges of image registration and tracking. We have been able to utilise SciKit-SurgeryBARD to perform rapid user studies examining the impact of visualisation design on depth perception [3] and on the occurrence of innattentional blindness [6] during augmented reality assisted surgery [2]. The lower part of Figure 1 shows an example from our innattentional blindness study, where the majority of our volunteers failed to notice the spider when performing a simulated surgical task. SciKit-SurgeryBARD enables the creation of lightweight applications using ArUco markers. For AR these are highly accurate as tracking and display use the same camera. This enables us to test different visualisation methods without worrying about tracking and registration accuracy.

Performance and Limitations: As a Python rather than a C++ library we are interested in the overall performance of the finished applications. We have observed that tasks such as instrument tracking and registration can be easily performed at the required frame rates ($\approx$ 30fps). One limitation we have observed is with rendering. With semi opaque anatomy the performance of the underlying VTK libraries implemted in Python is significantly degraded in comparison previous work using C++ [7]. To compensate for this we have to reduce the number of vertices in our anatomical models. For overlay accuracy down to 1mm this is not problematic, and indeed may be beneficial in terms of simplifying the display for the surgeon. Work is ongoing to determine the impact and cause of the reduced rendering performance.

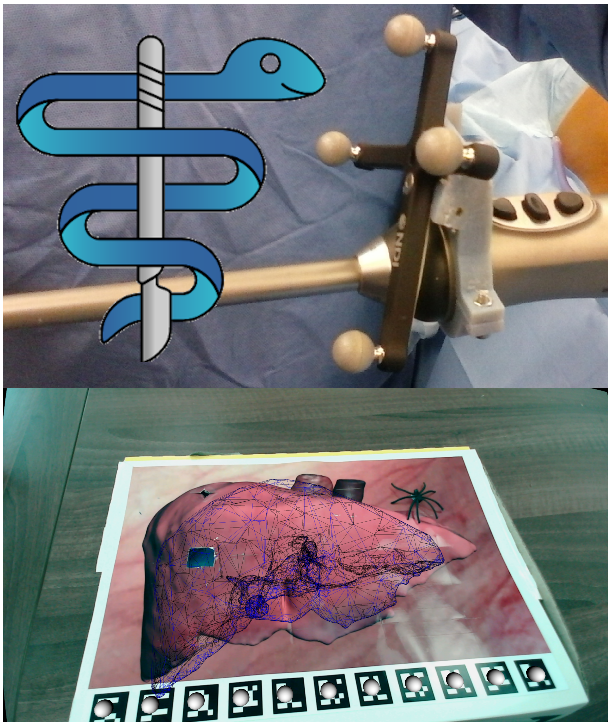

Fig. 1 Modular design is a key feature of Scikit-Surgery. Clinical hardware communicates through common interfaces defined in SciKit-SurgeryCore using Python's Abstract Base Class module. Modular design allows us to easily convert applications using clinical grade tracking hardware, NDI Polaris (at top), to applications suitable for visualisation and usability research beyond the lab. by utilising the open sourceArUco library (at bottom) and vice-versa.

DISCUSSION

The SciKit-Surgery libraries provide an open source set of tools to support researchers in basic research through to translational clinical projects. By publicly hosting the libraries on GitHub and engaging with the research community, we aim to encourage direct contribution from users, so that the libraries last beyond the funding cycle for an individual project.

By creating a set of loosely coupled libraries we have been able to focus development effort on those libraries that have gathered the most interest both internally and externally. We currently have over 40 libraries listed on the SciKit-Surgery organisation page. Given our resources it would not be possible to maintain them all, but as they are all largely independent this has not proven to be a problem.

Work is ongoing to better understand what practices lead to the most sustainable software and to better embed these practices into our development.

ACKNOWLEDGEMENT AND COPYRIGHT

This research was funded in whole, or in part, by the Wellcome Trust [203145Z/16/Z]. For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

REFERENCES

[1] S. Thompson, T. Dowrick *et al.*, "Scikit-surgery: compact libraries for surgical navigation," *International Journal of Computer Assisted Radiology and Surgery*, vol. 15, no. 7, pp. 1075–1084, Jul 2020. [Online]. Available: https://doi.org/10.1007/s11548-020-02180-5

[2] C. Davis, S. Yoo *et al.*, "Enhanced surgeons: Understanding the design of augmented reality instructions for keyhole surgery," in *2023 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, 2023, pp. 123–127.

[3] A. Reissis, S. Yoo *et al.*, "The effect of luminance on depth perception in augmented reality guided laparoscopic surgery," in *Medical Imaging 2023: Image-Guided Procedures, Robotic Interventions, and Modeling*, vol. 12466. SPIE, 2023, pp. 364–377.

[4] A. Hunt and D. Thomas, *The Pragmatic Programmer: From Journeyman to Master*. USA: Addison-Wesley Longman Publishing Co., Inc., 2000.

[5] S. Garrido-Jurado, R. Muñoz-Salinas *et al.*, "Generation of fiducial marker dictionaries using mixed integer linear programming," *Pattern Recognition*, vol. 51, 10 2015.

[6] A. S. Pandit, M. de Gouveia *et al.*, "Efficacy of a mindfulness-based intervention in ameliorating inattentional blindness amongst young neurosurgeons: A prospective, controlled pilot study," *Frontiers in Surgery*, vol. 9, 2022. [Online]. Available: https://www.frontiersin.org/article/10.3389/fsurg.2022.916228

[7] M. J. Clarkson, G. Zombori *et al.*, "The niftk software platform for image-guided interventions: platform overview and niftylink messaging," *International journal of computer assisted radiology and surgery*, vol. 10, pp. 301–316, 2015.