Software Design For A Robot To Assist Laparoscopic Surgery

Dayanne Fernandes da Cunha

Robotics and Automation Laboratory (LARA), University of Brasilia, Brazil
dayannefernandesc@gmail.com

Abstract. This paper presents a software design developed for a robot to assist laparoscopic surgery to be applied at Brazilian's public health system (SUS). It's a modular system expandable and able to be applied in multiple environments since powerful open source and cross-platforms frameworks/libraries as QT, Julius and OpenCV was used. The application of a Brazilian Portuguese acoustic model for voice recognition to control the robot is also a remarkable achievement.

Keywords: Software Design, Robotics, Laparoscopic Surgery, Speech Recognition

1 Introduction

In the beginning of 1980s, laparoscopic surgery was migrated from diagnostic to a surgical procedure. Semm K. and Muehe E. are the medics that introduced this technique for a wide field of indications, e.g., appendectomy, cholecystectomy, reflux surgery, gastric surgery, urology [1].

Nowadays the use of robotic surgery system are becoming more common, its benefits are being studied all around the world. Robotic technology allows the surgeon to increase dexterity and the degree of maneuverability to perform complex tasks in a minimally invasive fashion way [2].

Laparoscopic surgery demands an operating surgeon to make the medical procedure and a camera driver to show the surgeon the location of the operative field. This structure come with lots of problems, e.g. conflicts about the optimum visualization, fatigue of the assistant who holds the camera [3], inaccurate movements, image tremors.

There are many projects to solve the problems about the fatigue during laparoscopic surgery, e.g., EndoAssist [4], Vicky [5]. These solutions are very expansive to apply into Brazilian's public health system (SUS). In this context, CLARA is a low-cost project created at LARA (Robotics and Automation Laboratory, University of Brasilia) to help the operating surgeon to procedure the surgery.

CLARA's project was designed to help the operating surgeon to perform the procedure with full control of the tools through effective interfaces, voice recognition and a joystick linked with the laparoscopic grasper.

To be possible the application of the software into SUS, compatibility, modularity, fault-tolerance, security, usability, performance and maintainability were topics

considered during the software modeling process. This paper shows a software designed to assist laparoscopic robot surgery with all the aspects cited above.

2 Methods and Tools

Before the software modeling process, there were lots of visit at local public hospitals and some meetings with doctors that have procedure a laparoscopic surgery at least once. It was asked about the necessity of an automated robot to facilitate the surgery, and all the answers that we received was positive and supportive . The only concern on the analysis process was about the robot interfaces.

There is a study about the advantage of a voice control interface and a foot pedal interface, they concluded that voice control was more accurate and had the advantage of not requiring the surgeon to look away from the operative field [6]. Besides voice based control it was added a GUI (Graphical User Interface) and a Joystick interface.

A Brazilian Portuguese acoustic model [10] was used with Julius [8] (open-source speech recognition) to develop the voice control interface. Qt [7] was used to develop the GUI interface. Qt is used as the main framework of the system and C++11 as the main language tool. Qt is an open source, cross-platform framework with a thread library that helps control the asynchronous modules and a GUI component to produce its visualization.

The Joystick interface uses Gattlib to connect with a BLE (Bluetooth Low Energy). Commands send from the BLE device communicate with CLARA servomotors using sockets. A socket [11] is a generalized interprocess communication channel used to exchange messages between these processes running on different machines.

The servomotors are controlled with a Raspberry Pi 3. CLARA pass commands messages to the embedded system through ethernet cable using sockets. The Figure 1 shows the basic flow from one of the robot interface to the final stage, control the servomotors of the robot.

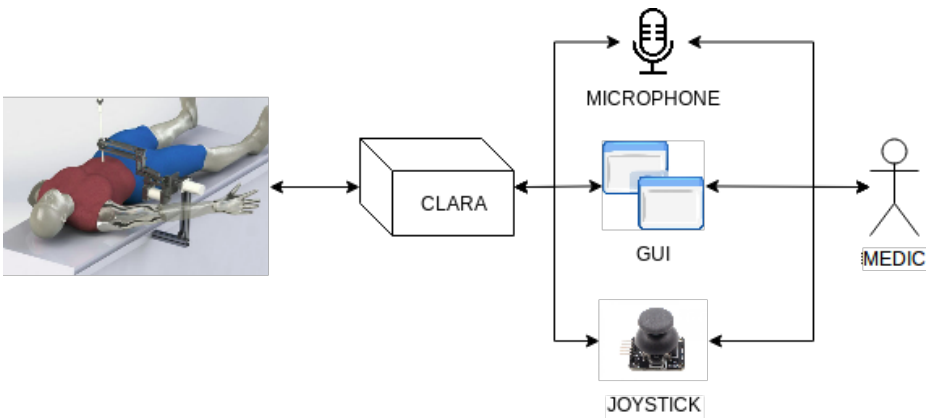

Figure 1: CLARA's interfaces to control the robot.

3 Results

An overview of CLARA software is illustrated at Figure 2. There are 8 modules with 5 threads (Vision, Speech, Motor, Joystick and the last one with UI, Control, Core and Network modules). Threads were used to make critical modules always available during the system execution since there are modules that capture data in real-time, e.g. Speech module.

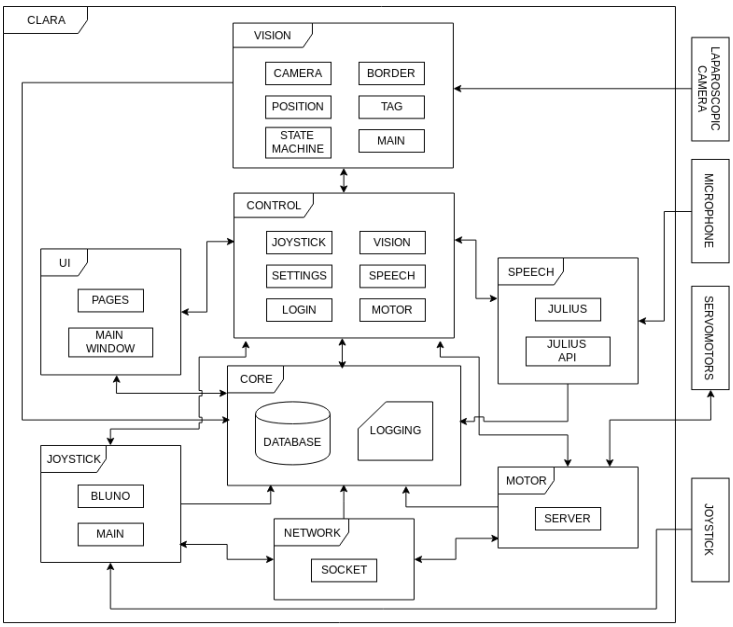

Figure 2: CLARA software design overview.

The usability and user experience was put as a priority to produce CLARA. In Figure 3 we can see the GUI basic usage flow from a login page (leftmost at first row) to the video page that shows images from the laparoscopic camera (rightmost at second row). The flow is consisted of log with an user, configure the system according the user preference, come back to the initial menu and then access video window to procedure with the surgery.

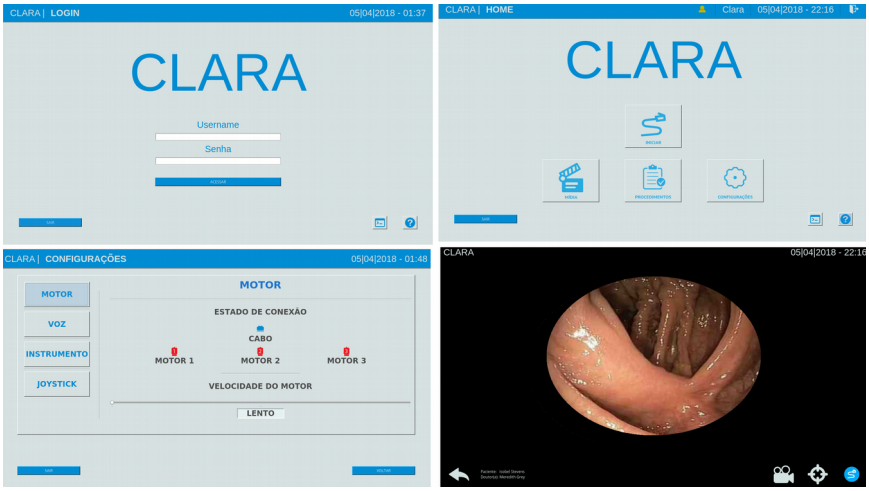

Figure 3: CLARA user interface.

The Vision module is responsible to estimate the 2D pose of a single instrument tip [9] using OpenCV and Boost's libraries. Using voice or GUI interface it's possible to centralize the laparoscopic camera at the grasper location in the human body.

For modules that uses sockets to communicate there are fault-tolerance to make secure tunnels connections through endpoints. CLARA is a system that has critical parts that cannot be off during the surgery, so it's necessary that critical modules is always available.

The motor client control sends to motor server control information about the connection of the 3 servomotors of the robot. Every second the server side supposed to receive the status of their connections. Above we list a solution to fault treatment in the network between server/client motor modules:

Motor server control:

1. Create socket, bind port and list to the port to communicate with the client (servomotors at Raspberry Pi);
2. While connection is not accepted between the systems, wait a second and try accept connection again;
3. After connection is setup, the system checks the stability of the connection every second;
4. If no data is received from the client socket in a second this means that the connection dropped, than a variable that informs the connection status is set to false, if any data is received than the connection status is set to true;
5. The event that runs every second to check the stability of the connection checks this status, if it's false the motor server is restart.

In motor client control there is a bash script that is automatically initialized when Raspberry Pi starts. The control flow is listed above:

1. In the main loop of servomotor control, for every iteration, check ethernet connection, if no connection is find than set variable connection status to false;
2. Inside of a thread, check if there is connection in the socket and if the cable ethernet is connected, block send/receive data if these two status are false;
3. Block the creation of the socket while no cable ethernet is connected;
4. If cable ethernet is connect than open socket in a port in Raspberry Pi IP;
5. If the socket could not be created than try again after five seconds;
6. If socket is down, reset the connection and try again in two seconds.

4 Conclusion

Develop a software to be applied in public hospitals in Brazil demands a few worries, it need to have a great user experience through easy and pleasant interfaces, open source libraries to decrease costs and some aspects that help future projects to use the code base to improve with more features.

In this paper CLARA modular design were shown. This type of software architecture improves topics like extensibility, maintainability and reuse of code. The voice, GUI and joystick interfaces help the users to communicate with the system with multiple ways, making the software more usable for many different types of users.

The fault-tolerance treatment, detailed in results, it's a important part of a robotic surgery system. Critical parts cannot fail during the procedure, so techniques are need to be implemented.

Security of the data passed between the core machine and the embedded system and performance of every module are future works to be add in the project.

5 Acknowledgements

CLARA project is funded by the Brazilian Ministry of Health (Term of Cooperation no 121/2013, Process 25000.169843/2013-83).

I would like to express my deep gratitude to tenured assistant professor Mariana Bernardes and Professor Geovany Araújo for their valuable and constructive suggestions during the planning and development of this research work. Advice given by Tomaz Canabrava about QT e C++ tools has been a great help in this project.

References

1. Buia, A., Stockhausen, F., Hanisch, E.: Laparoscopic surgery: A qualified systematic review. World J Methodol, 5(4):238–254 (2015).
2. Singh, I.: Robotics in urological surgery: Review of current status and maneuverability, and comparison of robot-assisted and traditional laparoscopy. Computer Aided Surgery, 16(1):38–45 (2011). PMID: 21198426.
3. Liu, S., Hemming, D., Luo, R. B., Reynolds, J., Delong, J. C., Sandler, B. J., Jacobsen, G. R., Horgan, S.: Solving the surgeon ergonomic crisis with surgical exosuit. Surg Endosc, 32(1):236–244 (2018).
4. Kommu, S. S., Rimington, P., Anderson, C., Rane, A.: Initial experience with the EndoAssist camera-holding robot in laparoscopic urological surgery. J Robot Surg, 1(2):133–137 (2007).
5. Long, J. A., Cinquin, P., Troccaz, J., Voros, S., Berkelman, P., Descotes, J. L., Letoublon, C., Rambeaud, J. J.: Development of miniaturized light endoscope-holder robot for laparoscopic surgery. J. Endourol., 21(8):911–914 (2007).
6. Wagner, A. A., Varkarakis, I. M., Link, R. E., Sullivan, W., Su, L. M.: Comparison of surgical performance during laparoscopic radical prostatectomy of two robotic camera holders, EndoAssist and AESOP: a pilot study. Urology, 68(1):70–74 (2006).
7. Nokia Corp. Qt : cross-platform application and ui framework (2012).
8. Akinobu, L., Tatsuya, K.: Recent development of open-source speech recognition engine julius. In Proceedings: APSIPA ASC: Asia-Pacific Signal and Information Processing Association, Annual Summit and Conference, International Organizing Committee, 131–137 (2009).
9. Vergara, R. F., Miranda, G. M., Perruci, P. H. S., Bernardes, M. C.: Edges based surgical instrument tracking in laparoscopic images. XXV Brazilian Congress on Biomedical Engineering, 1223–1226 (2016). ISSN: 2359-3164.
10. Neto, N., Patrick, C., Klautau, A. et al. J Braz Comput Soc (2011) 17: 53. https://doi.org/10.1007/s13173-010-0023-1.
11. GNU. Sockets, http://kirste.userpage.fu-berlin.de/chemnet/use/info/libc/libc_11.html. Last accessed 05 April 2018.