2014 5th IEEE RAS & EMBS International Conference on
Biomedical Robotics and Biomechatronics (BioRob)
August 12-15, 2014. São Paulo, Brazil

Command Interface and Driving Strategy for a Voice Activated Endoscope Positioning Arm*

Eric T. P. Santos, Idágene A. Cestari, *Member, IEEE*

Abstract— **A command and control interface for intraoperative endoscope positioning must be intuitive, simple, intrinsically safe and reliable. Voice commands are widely used in commercial and experimental robotic-assisted remote-controlled surgical systems and must comply with those requirements. This paper presents a design proposal and implementation of an integrated voice-activated control interface as well as its associated command strategy. It comprises an isolated word speech recognition module based on the IBM SMAPI programming library, a task management state machine and a RS232 communication module.**

I. Introduction

Traditional open surgical procedures are performed through large incisions into the surgical area of interest. The quest to reduce the associated trauma led to the development of minimally invasive surgical techniques performed through considerably smaller accesses. Surgical advancements have been pushed by technological improvement of surgical instrumentation [1], continuous reduction on the size of incisions [2] and robot-assisted surgical systems, which typically contain two or three robotic arms for instrument positioning and another for endoscope positioning. They are usually designed for a remotely controlled setup, where the surgeon commands the arms from a distance, usually from a control station in the OR or in an adjacent room, but possibly from another geographical location.

Minimally invasive surgery may reduce recovery times (and, thus, costs of hospital stay), infection rates, postoperative pain and bleeding [3, 4]. It may, however, be limited by increased operating times [5], difficult instrument handling, restricted intraoperative movements, reduced tactile feedback and lack of depth visual perception [6].

The command interface for the endoscope-positioning arm should be designed to keep surgeon's hands free for instrument manipulation, while simultaneously offering reliable communication to the underlying hardware. Voice commands are often used [7, 8], although different approaches, like head motion sensors [9, 10], joysticks [11] and pedals, are sometimes also implemented.

This article describes an isolated word speech recognition interface for controlling an endoscope-positioning arm as well as the underlying command strategy, considering the relevant safety requirements for minimally invasive robot-assisted surgical procedures.

*Research funded by FAPESP (São Paulo Research Foundation).

E. T. P. Santos, Department of Electric Engineering, Polytechnic School, University of São Paulo.

I. A. Cestari. Bioengineering Division, Heart Institute (InCor), University of São Paulo (phone: +55 11 2661 5522, email: cestari@incor.usp.br).

II. Implementation

The command interface described herein has been designed to drive a robotic arm adapted to endoscope positioning, designed by the Special Robots Laboratory (Department of Mechanical Engineering, University of São Paulo) [12]. The prototype is driven by step motors and is based in the "parallel bars" mechanical model [11], shown in Fig. 1.

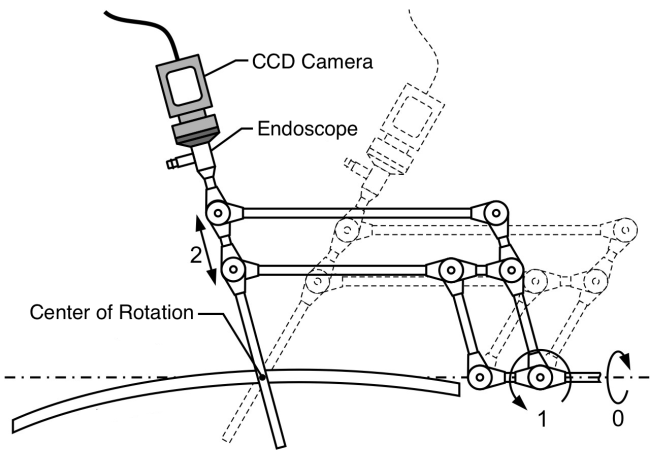

Figure 1. Parallel bars mechanism, a 3 DOF robotic model with a center of rotation.

It is a 3 degrees-of-freedom mechanism, with intrinsic mechanical restrictions, so that any possible trajectory is executed around a "center of rotation". When this point coincides with the endoscope insertion point through the skin, the translational forces on the soft tissue are greatly reduced, thus minimizing the risk of tissue damage around the incision.

The voice command interface was designed to enable fast and accurate arm movement along all degrees of freedom. The proposed solution has three main subsystems, as shown in Fig. 2 and described in further detail in the following sections.

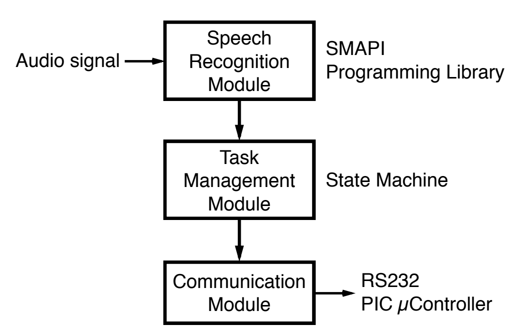

Figure 2. Voice command interface main subsystems.

A. Speech Recognition Module

SMAPI (Speech Manager Application Programming Interface) is a programming library developed by IBM for real-time speech recognition in speaker-dependent and speaker-independent modes. It's distributed as a dynamic link library (DLL), loaded in runtime by a caller code. It takes an audio data stream as input, decodes it, recognizes the utterance against a predefined vocabulary and returns a text string containing the recognized word. It's the application's responsibility to take action based on the received text, execute corresponding tasks, and provide exception-handling code if SMAPI cannot recognize the spoken word.

An audio input module acquires audio signal from microphone port at 22 kHz, 16 bit. The resulting data is sent to a acoustic preprocessor which optimizes the signal and sends it to a stochastic recognition algorithm.

SMAPI allows the definition of multiple *dynamic vocabularies* (word lists, provided to the library as arrays of text strings) that can be activated or deactivated in runtime. The recognition algorithm compares the acquired audio signal with a set of phonetic and linguistic features extracted from each word in the active vocabulary and chooses the one most likely matching the recorded sound. If, however, the matching probability is under an adjustable rejection threshold, the utterance is ignored and a false-negative exception is raised.

B. Task Management Module

The task management module has been designed to improve intraoperative safety. It contains a state machine that receives the word identified by the speech recognition module, processes it and activates or deactivates the dynamic vocabularies. State transitions also trigger vocabulary switching to reduce the number of valid commands at any given time, minimize the number of false-positive errors and avoid unintended movement.

The complete command set contains nine words split among three different vocabularies, as shown in Table 1. Each word is associated to one specific state transition of the task management module. Vocabulary I corresponds to every possible movement of the mechanism along its three degrees of freedom plus a "sleep" command, intended to put the mechanism in an inactive state ("sleep mode"). It's impossible to initiate arm movements in this state for increased safety. Vocabulary II contains the stop-action, the only valid command during arm movement, and Vocabulary III contains the "wake-up" command valid during sleep mode.

TABLE 1: DYNAMIC VOCABULARIES

I	II	III
"Up"	"Stop"	"Wake Up"
"Down"		
"Left"		
"Right"		
"Forward"		
"Back"		
"Sleep"		

C. Communication Module

Acts as a bridge between the task management module and a PIC16C773 microcontroller (Microchip Technology Inc., Chandler, Arizona, USA) that actually sends the driving signals to the mechanism. It was designed for communication simplicity, speed and scalability. It uses a custom communication protocol with 1-byte instructions corresponding to each state transition in the task management module. Instructions are then transmitted over RS232, received and decoded by the microcontroller, which then generates the electrical sequences to drive the step motors on each joint.

III. RESULTS

Fig. 3 shows the architecture diagram for the speech recognition module as currently implemented. It shows SMAPI as an external library that takes its configuration parameters (including vocabulary definition) from a specific management module and returns messages that are handled by a callback function for processing. Returned messages usually contain the recognized word as a text string, as well as the estimated match probability and, eventually, an error code. In case of unmatched expressions (i.e. with match probability under the rejection threshold), the message contains a "blank" recognized word. In that case, specific exception handling functions are called.

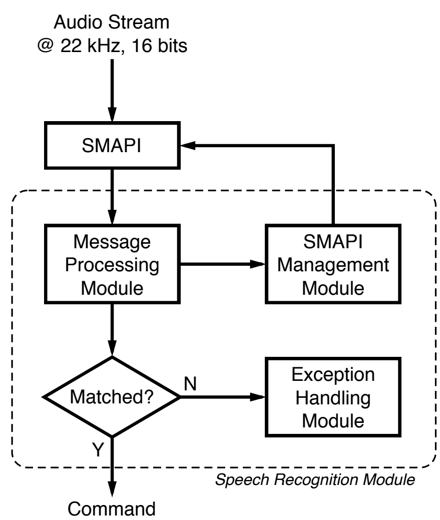

Figure 3. Architecture diagram for the speech recognition algorithm.

Fig. 4 shows the architecture diagram for the task management and serial communication modules. Messages received from the speech recognition algorithm are processed by a decision routine to determine if the matched word is a valid command. If so, the received command is sent to the state machine. Besides managing the state transitions, it informs the speech recognition module which vocabulary should be activated in each state. The selected command is then sent to the communication module, which generates the instructions that are sent to the microcontroller through the serial interface. Otherwise, exceptions are handled by a specific function and no state transition occurs.

Fig. 5 shows the state diagram for the state machine and which specific words trigger corresponding state transitions. It's important to emphasize that there's only one dynamic vocabulary active at any given state and that no state transition occurs when SMAPI returns a blank recognized command (not recognized or false negative), except if the arm is moving, in which case it immediately stops. Some state transitions are also triggered by timers T1, which stops the arm after 3 seconds if no other command is issued during any arm movement, and T2, which puts the system in sleep mode if the arm is not moving and no valid command is received for 30 seconds.

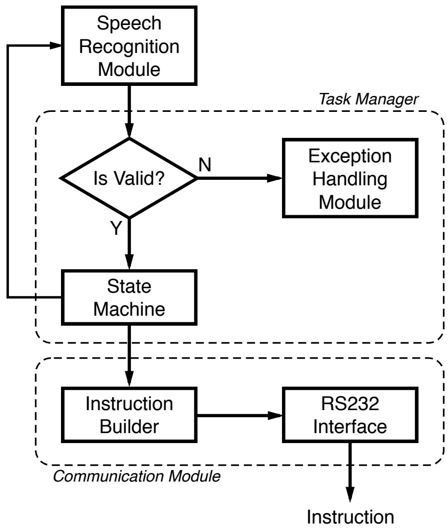

Figure 4. Architecture diagram for the task management and serial communication modules.

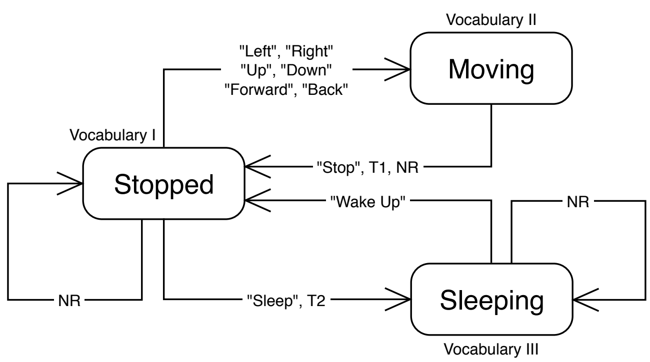

Figure 5. State diagram for the state machine in the task management module. T1 = timer 1, T2 = timer 2, NR = not recognized.

The instruction builder can assemble four different instructions that correspond to the state transitions as defined by the state machine and also take into account which joint (motor) has to be moved and in which direction. The instructions are then sent through the serial communication interface using the RS232 standard port. Fig. 6 shows low-level binary format for each available instruction.

IV. Test Protocol

A test protocol was designed to evaluate software functionality. The voice commands of 5 different speakers were recorded and saved locally as 22 kHz, 16 bits, uncompressed audio files. Each of the nine available commands was repeated 10 times by each speaker in random order to capture subtle variations in utterance and voice tone. The SMAPI library was then trained in speaker-dependent mode for each speaker using the built-in training functionality.

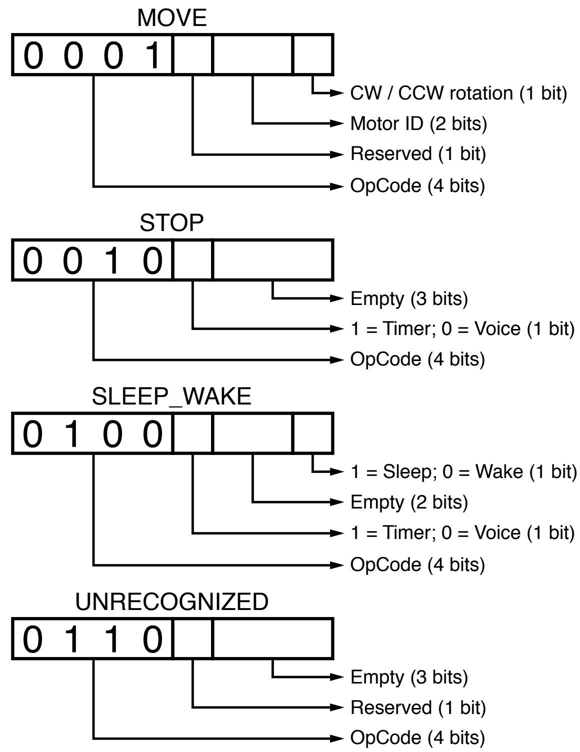

Figure 6. Binary format of the instruction set assembled by the instruction generator. The labels above each instruction represent their names.

Fig. 7 shows an architecture diagram of the test environment. A dedicated test unit was implemented to load the audio files from disk in a predefined activation sequence, taking into account the current system state (and, therefore, the currently active vocabulary). The command interface processes and attempts to match each reproduced audio signal. When the commands are recognized, corresponding instructions are sent to the microcontroller. An instruction parser then records the output signals of the microcontroller to the mechanism.

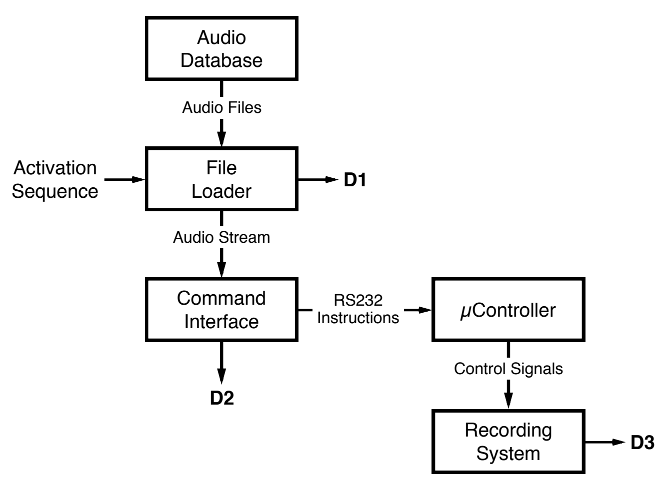

Figure 7. Architecture diagram of the test environment.

The test protocol generates three data files D1, D2 and D3. The file loader records the reproduced command (D1). The command interface records the recognized word (D2) and the instruction parser records the output signal (D3). The matching rate for the speech recognition module is calculated by comparing D1 and D2. The accuracy rate of the communication interface is calculated by comparing D2 and D3

V. CONCLUSION

Voice commands are simple, reasonably intuitive, have short learning curves, can be easily adapted to different activation protocols and keeps surgeon's hands free at all times. Our results show that safety and reliability requirements may be achieved by properly managing system states or by including confirmation commands, such as those used by Slate *et al* [7].

We present herein the development of an isolated word voice command interface for robotic-assisted endoscope positioning that allows the user to keep hands free during the entire surgical procedure. The development of the specific command strategy was based on previous work [13]. Our results suggest that the dynamic vocabulary switching, the use of a state machine and the inclusion of specifically tailored exception handling functions contribute to improve intrinsic system safety and reliability.

Once the interface effectiveness has been demonstrated, further improvements in the command strategy are possible and include position recording for faster return to previous saved locations and surgeon profiles to further enhance recognition accuracy, among others.

REFERENCES

[1] M. O. Schurr, A. Arezzo, G. F. Buess, "Robotics and systems technology for advanced endoscopic procedures: experiences in general surgery", *European Journal of Cardio-thoracic Surgery*, vol. 16, no. 2, pp. S97-S105, 1999.

[2] J. E. Felger, L. W. Nifong, W. R. Chitwood, "The evolution of and early experience with robot-assisted mitral valve surgery", *Surgical Laparoscopy, Endoscopy and Percutaneous Techniques*, vol. 12 no. 1, pp. 58-63, 2002.

[3] J. A. Morgan, *et al*, "Robotic techniques improve quality of life in patients undergoing atrial septal defect repair", *Annals of Thoracic Surgery*, vol. 77, no. 4, pp. 1328-1333, 2004.

[4] J. Rosen, *et al*, "Force controlled and teleoperated endoscopic grasper for minimally invasive surgery: experimental performance evaluation", *IEEE Transactions on Biomedical Engineering*, vol. 46, no. 10, pp. 1212-1221, 1999.

[5] R. Kolvenbach, *et al*, "Total laparoscopically and robotically assisted aortic aneurysm surgery: A critical evaluation", *Journal of Vascular Surgery*, vol. 39, no. 4, pp. 771-776, 2004.

[6] V. Falk, *et al*, "Quality of computer enhanced totally endoscopic coronary bypass graft anastomosis – comparison to conventional technique", *European Journal of Cardio-thoracic Surgery*, vol. 15, pp. 260-265, 1999.

[7] A. P. Slade, *et al*, "An integrated control structure for surgical assist robotics for laparoscopy", presented at the XIV Brazilian Congress of Mechanical Engineering, São Paulo, SP, Brazil, 1997.

[8] V. F. Muñoz, *et al*, "Design and Control of a Robotic Assistant for Laparoscopic Surgery", presented at the 9th International Symposium on Intelligent Robotic Systems, Toulouse, France, pp. 18-20, 2001.

[9] E. Kobayashi, *et al*, "A New Safe Laparoscopic Manipulator System with a Five-bar Linkage Mechanism and an Optical Zoom", *Computer Aided Surgery*, vol. 4, pp. 182-192, 1999.

[10] P. A. Finlay, M. H. Ornstein, "Controlling the Movement of a Surgical Laparoscope", *IEEE Engineering in Medicine and Biology*, vol. 14, no. 3, pp. 289-291, 1995.

[11] R. H. Taylor, *et al*, "A Telerobotic Assistant for Laparoscopic Surgery", *IEEE Engineering in Medicine and Biology*, vol. 14, no. 3, pp. 279-288, 1995.

[12] W. B. Vidal Filho, "Desenvolvimento de uma estação robótica para cirurgias minimamente invasivas", PhD thesis, Dept. Mechanical Engineering, University of São Paulo, 2003.

[13] E. T. P. Santos, "Sistema de Reconhecimento de Comandos Isolados de Voz para Aplicação em Robótica Cirúrgica", M.Sc. thesis, Dept. Electrical Engineering, University of São Paulo, 2003.