# Design and Sensor-based Control of a Motorised Endoscope Manipulator for Sinus Surgery

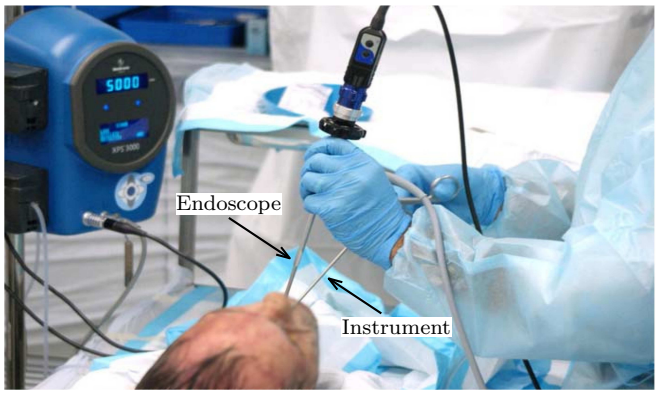


Fig. 1. In traditional FESS, the surgeon directly manipulates the endoscope.

***Abstract*—abstract**

***Index Terms*—Robot manipulators, endoscopic sinus surgery, remote centre of motion, motion control, human-machine interface.**

## I. Introduction

See Fig. 1.
[1], [2]

### A. Related Work

### B. Contribution

### C. Organisation

The rest of this paper is organised as follows: Section II describes the design of the manipulator; Section III derives the control algorithms; Section IV presents the conducted experiments; Section V gives final conclusions.

## II. Mechanism Design

In this section, we present and analyse the structure of the proposed endoscope manipulator.

### A. Task Requirements

In traditional endoscopic sinus surgery, the surgeon first inserts the endoscope camera inside the patient's nostril, which serves as a natural pivot point. With one hand, the surgeon then manipulates the endoscope to observe different regions of interest inside the nasal cavity. This type of motion consists of two rotations (which control the camera's pan and tilt motions)

This work is supported in part by the Hong Kong RGC under grant 415011 and grant CUHK6/CRF/13G, and in part by the Hong Kong ITF under grant ITS/020/12FP.

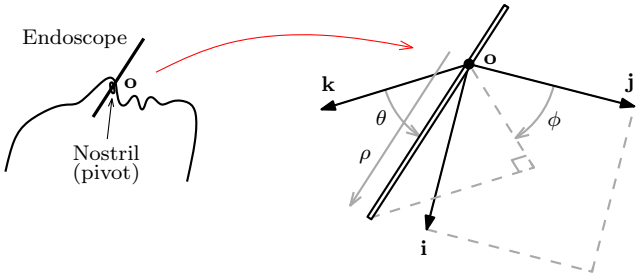


Fig. 2. Constrained motions of the endoscope, where $\phi$ and $\theta$ represent its orientation, and $\rho$ represents its insertion distance.

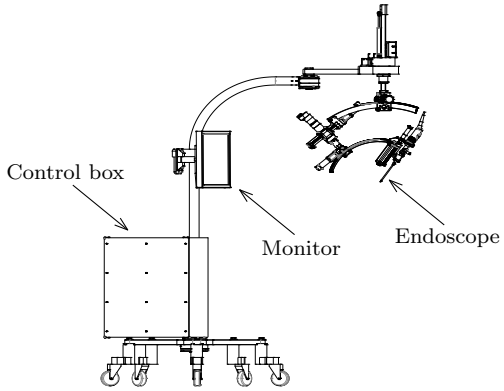


Fig. 3. Conceptual model of the complete robotic system.

and one linear translation (which controls the camera's zoom motion). The camera's roll angle (i.e. the rotation around the optical axis) is typically kept constant throughout the operation, therefore, we intentionally exclude it from our analysis. See Fig. 2 for a conceptual representation of the camera's DOF.

To perform the motions that are necessary to conduct endoscopic sinus surgery, we propose to develop a new robotic system composed of a 5-DOF passive positioning mechanism and a 4-DOF active manipulator. The former part allows the surgeon to manually position the system's RCM at the patient's nostril during the set-up stage, the latter part allows the surgeon to actively command the motion of the camera. The whole mechanism is placed on a wheeled mobile platform which also houses the motion control system. See Fig. 3 for a conceptual model of the proposed endoscope manipulator.

### B. Passive Positioning Mechanism

This mechanism has five passive degrees of freedom: four revolute joints and one prismatic joint. The first four joints $q_1, \ldots, q_4$ form a SCARA-like kinematic structure [3]. The structure of the fifth joint $q_5$ is a arc gear-rack mechanism

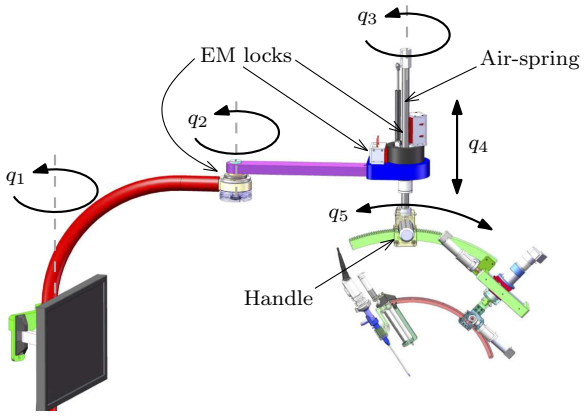


Fig. 4. The passive positioning mechanism.

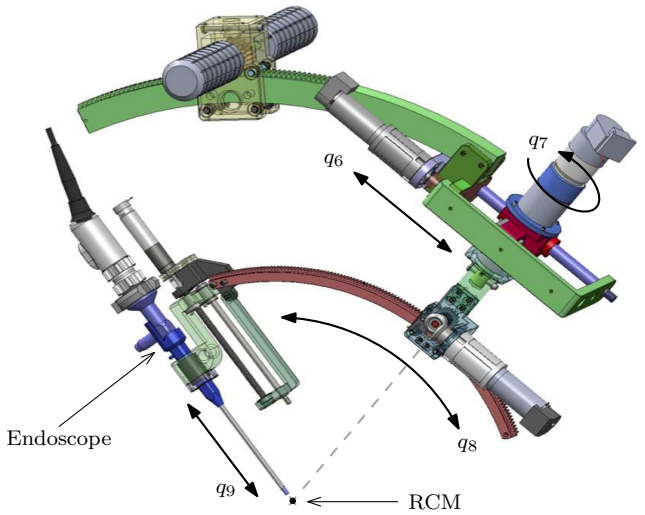


Fig. 5. The active endoscope manipulator.

that rotates the active manipulator. See Fig. 4 for a 3D model of this mechanism.

The purpose of the SCARA-like structure is to allow the surgeon to manually position the active system in the 3D space during set-up; this structure can also rotate the system along the vertical axis. The purpose of the arc joint $q_5$ is to allow the surgeon to set with a manual handle the desired 'home' orientation for the motorised tilting angle of the camera (to be described in the next section)

The vertical joint $q_4$ provides linear up-and-down motions to the system. This mechanism is gravity-balanced so that it can be easily positioned by the surgeon during set-up. To implement this behaviour, we use the following components: an air-spring, a spline axis, a damper, and an electromagnetic (EM) lock. The air-spring can support a load of nearly 80N; we incorporate a damper to stabilise the force of this component.

Once the endoscope camera has the desired position and orientation inside the patient's nasal cavity, the passive positioning mechanism can be fixed with a series of EM locks that are embedded into the joints $q_2$, $q_3$, and $q_4$. These breaks are activated/deactivated with a simple on/off switch.

### C. Active Endoscope Manipulator

The structure of the active manipulator is depicted in Fig. A. This system has four motorised degrees of freedom, two prismatic joints and two revolute joints, all driven by DC motors. The prismatic joints $q_6$ and $q_9$ have a ball-screw structure; the joint $q_7$ directly rotates an arc gear-rack mechanism that is driven by joint $q_8$.

The purpose of the linear joint $q_6$ is to translate the RCM point into its desired location (i.e. inside the patient's nostril). For that, the active manipulation most be first manually adjusted with the passive mechanism such that the axis of $q_6$ is parallel to the sinus cavity. Note that this motorised joint is not used very often during the operation; we only use it during set-up or whenever there is the to readjust the RCM point (e.g. when changing the patient's posture).

The purpose of the last three joints $q_7$, $q_8$ and $q_9$ is to control the pan, tilt, and zoom motion of the endoscope camera, respectively. The axes of motion of the rotational joints $q_7$ and $q_8$ are designed such that they are always orthogonal, and the intersection of its axes determines the RCM point of the manipulator. To prevent injures resulting from large motions, the fixed RCM point must be set at the entry of the patient's nostril.

### D. Kinematics

From Fig. 2, we can see that the constrained 3-DOF configuration of the endoscope can be locally represented with spherical coordinates. The specific design of the active endoscope manipulator allows us to control this configuration directly with the last three joints. To this end, we must first define the 'home' (i.e. zero) configuration by rotating joint $q_8$ such that the axes $q_6$ and $q_9$ are parallel. With this reference configuration, the joint $q_7 = \phi$ represents the angle measured over the plane $(\mathbf{i}, \mathbf{j})$, the joint $q_8 = \theta$ represents the angle between the endoscope and the axis $\mathbf{k}$, and the joint $q_9 = \rho$ represents the insertion distance.

We can compute the Cartesian position vector $\mathbf{p} \in \mathbb{R}^3$ of the endoscope's tip with the following expression:

$$\mathbf{p} = \rho \begin{bmatrix} \sin(\theta)\cos(\phi) & \sin(\theta)\sin(\phi) & \cos(\theta) \end{bmatrix}^{\mathsf{T}}. \quad (1)$$

The coordinates of the vector $\mathbf{p}$ are defined with respect to the RCM point $\mathbf{o}$.

## III. Controller Design

In this section, we describe the developed control system and propose hand-free methods to control the position of the manipulator.

### A. Motion Control System

The motion control system that we developed for this robot is composed of a low-level industrial motion controller, and a high-level PC-based controller. As the low-level controller, we use a Galil DMC-1440 motion servo-controller; this embedded board decodes the motors' positions and outputs the current control signals (calculated by an inner PID algorithm) that drive the DC motors. As the high-level controller, we use an

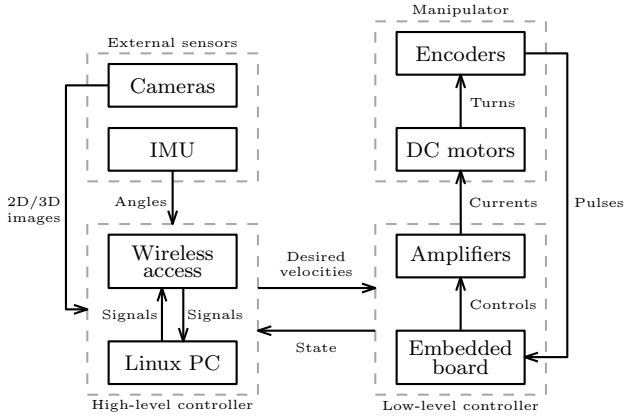


Fig. 6. Schematic representation of the motion control system.

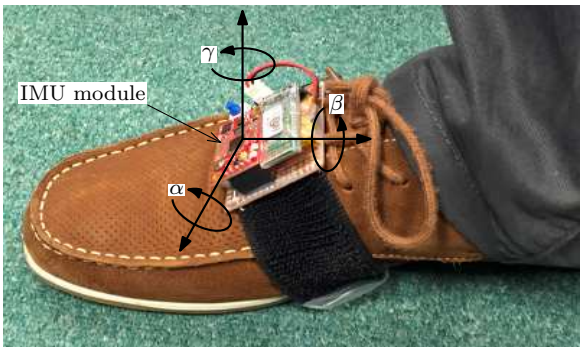


Fig. 7. The foot-controlled interface with its orientation angles.


Fig. 8. The gesture to enable/disable the foot-controlled interface.

industrial PC (i5-3550S CPU) running a Linux-based operative system; this PC processes all external sensor feedback and computes the motion control algorithms. The communication between the low-level and high-level modules is done via high-speed Ethernet. All the motorised joints of the manipulator are controlled in velocity-mode, with a servo-loop of 10 milliseconds. See Fig. 6 for a schematic representation.

To guarantee the absolute safety during the surgical procedure, the control system must operate with a strict (i.e. deterministic) real-time behaviour. We provided this valuable feature to our high-level controller by installing a Linux kernel with a customised real-time patch. For that, we use the Xenomai real-time development framework, see [4] for details.

To develop the hand-free user interface for the motorised manipulator, we integrate different sensors into our system. We use an inertial measurement unit (IMU) to measure the orientation/posture of the user's body; the measured angles from this device are processed with a microcontroller board (Arduino Uno) and are wirelessly transmitted to the PC with Bluetooth device at 57600 bps. To process in real-time the captured images from the endoscope camera, we use a digital TV-card (Hauppauge) with USB interface. The design of these control interfaces is described in the following sections.

### *B. Foot-controlled Interface*

The purpose of our robotic manipulator is to enable the surgeon to conduct two-hand operations (i.e. to simultaneously manipulate two instruments) while retaining direct control over the endoscope camera. To achieve this objective, we developed an IMU-based interface that uses foot gestures to control the camera motions. This device is attached to the user's foot, as shown in Fig. 7.

With the IMU device, we measure the orientation angles $\alpha, \beta, \gamma \in \mathbb{R}$ of the user's foot in real-time; we use these feedback angles to control the last three motorised joints (i.e. the pan, tilt, and zoom joints) of the manipulator. Note that the control of the endoscope's position with a foot gesture is a difficult task because the motion of the foot is not as dexterous as the motion of a hand. For this reason, with our proposed foot interface we only control one joint of manipulator at the time and with a constant speed command. The developed interface has implemented the following actions: (1) enable/disable the control interface, (2) select the active joint, and (3) command the forward/backward motion. Fig. 8 and Fig. 9 graphically depict the implemented foot gestures.

The control logic of this interface is detailed in the following pseudocode:

**loop** ▷ Main loop of the foot-controlled interface
  **if** $\alpha < 10° \rightarrow \alpha > 60° \rightarrow \alpha < 10° \rightarrow \alpha > 60°$ **then**
    Enable / disable interface
  **end if**
  **if** Interface enabled **then**
    **if** $\alpha < 10°$ and $(\beta < -15° \rightarrow \beta > 15°)$ **then**
      Change active joint
    **else if** $10° \leq \alpha \leq 60°$ and $|\gamma| > 15°$ **then**
      Move active joint in the direction of $\text{sgn}(\gamma)$
    **else if** No command for 10 seconds **then**
      Disable interface
    **end if**
  **end if**
**end loop**

(explain with words this algorithm)

In this algorithm we use the symbol $A \rightarrow B$ to represent the transition from state $A$ to state $B$. Note that the threshold parameters (e.g. $10°$) can be calibrated during set-up depending on the user requirements. The "change active joint" command incrementally shift among the last three motorised joints, i.e. from $q_7 \rightarrow q_8 \rightarrow q_9 \rightarrow q_7 \rightarrow \cdots$ and so on. To facilitate the operation of the interface, the commands are automatically verbalised by the system as follows: "enabled", "disabled", "pan", "tilt", and "zoom".

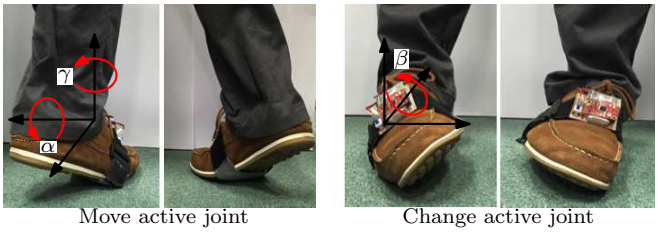


Fig. 9. The foot gestures to (left) command forward/backward motion to the joint and (right) change the current active joint.

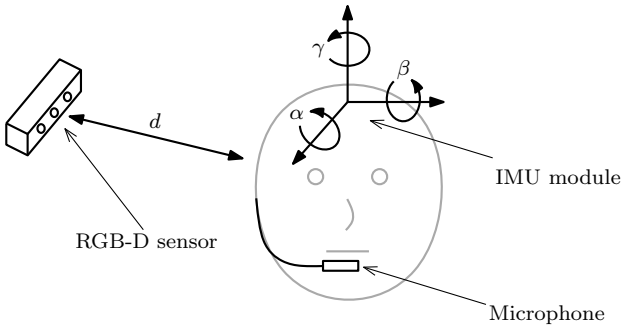


Fig. 10. The head-controlled interface with orientation angles, and microphone.

### C. Head-controlled Interface

In this section we describe the design of an interface that uses head motions to control the endoscope camera. As with the previous approach, we use an IMU module to measure the orientation of the head. We additionally integrate an RGB-D sensor (Kinect) to measure 3D displacements of the head, and a wireless microphone to allow the user to use voice commands. The developed head-controlled interface is depicted in Fig. 10.

By using head motions, we can implement a more natural relation between the gestures of the user and the motion of the camera. In our approach, we use the "up/down" angle $\beta$ to control the camera's tilting (i.e. vertical) motion, and the "left-right" angle $\gamma$ to control the camera's pan (i.e. horizontal) motion. In our set-up, the RGB-D sensor is placed on top of the surgeon's monitor, therefore, we can measure the scalar distance $d$ between the head and the monitor in real-time; we use this relative distance (or proximity) to control the zooming motions of the camera. To enable/disable the interface, two simple voice commands are implemented: "enable control" and "stop control". We use speech recognition algorithms (CMU Pocketsphinx libraries) to automatically detect these commands. For safety reasons, only one joint of the robot can be controlled at the time. Fig. 11 depicts the implemented head gestures.

The control logic of the interface is detailed in the following pseudocode:

**loop** ▷ Main loop of the head-controlled interface
  **if** "enable/stop control" command is detected **then**
    Enable/disable interface
  **end if**
  **if** Interface enabled **then**
    **if** $|\gamma| > 30°$ **then**
      Move joint $q_7$ in the direction of $\mathrm{sgn}(\gamma)$
    **else if** $|\beta| > 30°$ **then**
      Move joint $q_8$ in the direction of $\mathrm{sgn}(\beta)$
    **else if** $|d - d_0| > 10$cm **then**
      Move joint $q_9$ in the direction of $\mathrm{sgn}(d - d_0)$
    **else if** No command for 10 seconds **then**
      Disable interface
    **end if**
  **end if**
**end loop**

Fig. 11. The head gestures to command the motion of the endoscope manipulator.

The purpose of this method is to provide the surgeon with an intuitive way to control the camera: e.g. by rotating the head to the left the camera then rotates to the left, or by approaching to the monitor the camera zooms-in. Similarly to the previous interface, the threshold parameters can be calibrated during the system's set-up depending on the user's requirements.

### D. Automatic Image-based Instrument Tracking

In this section we present an uncalibrated controller to automatically track the instrument manipulated by the surgeon. To achieve this objective, we integrate into the control system the image feedback captured by the endoscope camera. The purpose of this approach is to automatically move the instrument to the centre of the captured scene, which allows the surgeon to observe specific areas inside the nasal cavity. Similarly to the head-controlled interface, we also use a microphone to enable this auto-track control modality.

The design of the controller is based on captured images of the surgical instrument, see Fig. 12 for a conceptual representation of the set-up. Let us define the *imaged* tip of the instrument by $\mathbf{m} \in \mathbb{R}^2$. The desired image position of the tip is defined by $\mathbf{m}_d \in \mathbb{R}^2$, which in our application represents the centre coordinates of the capture image. With this controller, we only consider the motion of the pan and tilt joints, i.e. joints $q_7$ and $q_8$. For ease of presentation, we define the input velocity vector $\dot{\mathbf{q}} \in \mathbb{R}^2$ as follows:

$$\dot{\mathbf{q}} = \begin{bmatrix} \dot{q}_7 & \dot{q}_8 \end{bmatrix}^\intercal \tag{2}$$

From standard visual servoing [], we know that there exists a differential kinematic relation between the input motion $\dot{\mathbf{q}}$ of the moving camera and the measured optical flow $\dot{\mathbf{m}}$ of the image point. We model this differential relation as follows:

$$\dot{\mathbf{m}} = \mathbf{J}\dot{\mathbf{q}} \tag{3}$$

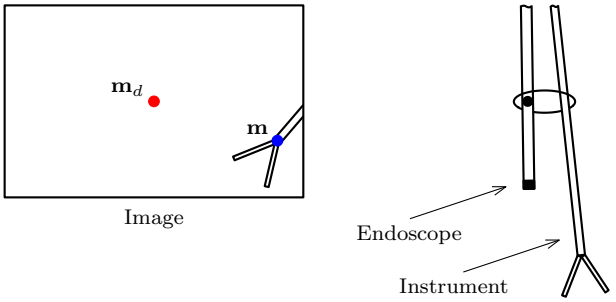


Fig. 12. Conceptual representation of the image-based instrument tracking control.

where the square matrix $\mathbf{J} \in \mathbb{R}^{2\times 2}$ denotes the image Jacobian matrix, which depends on the camera's calibration parameters and the 3D Cartesian position of the instrument. Note that since the instrument is manipulated by the surgeon, its 3D position vector is not known to us and is difficult to estimate using 2D images from a monocular camera. Therefore, the matrix $\mathbf{J}$ cannot be analytically computed. To cope with this issue, in this work we propose to compute an adaptive estimation, here denoted by $\widehat{\mathbf{J}} \in \mathbb{R}^{2\times 2}$, of the unknown Jacobian matrix.

Consider the following definition of the scalar functional

$$H = \frac{1}{2}\left\|\widehat{\mathbf{J}}\dot{\mathbf{q}} - \dot{\mathbf{m}}\right\|^2 \tag{4}$$

which quantifies the accuracy of the estimated matrix $\widehat{\mathbf{J}}$. In our method, we vary the elements $\widehat{J}_{ij} \in \mathbb{R}$ of this matrix with the gradient descent rule

$$\frac{\mathrm{d}}{\mathrm{d}t}\widehat{J}_{ij} = -\lambda\frac{\partial H}{\partial \widehat{J}_{ij}} \tag{5}$$

where the scalar $\lambda \in \mathbb{R}$ represents a positive tuning gain; Appendix A presents the stability analysis of this adaptive estimator. With our uncalibrated adaptive method, we compute the velocity control input as

$$\dot{\mathbf{q}} = -k\widehat{\mathbf{J}}^{-1}\Delta\mathbf{m} \tag{6}$$

where $k \in \mathbb{R}$ denotes a positive (proportional-like) feedback gain, and $\Delta\mathbf{m} = \mathbf{m} - \mathbf{m}_d \in \mathbb{R}^2$ denotes the image error.

The following pseudocode details the implementation of our method

## IV. Experiments

In this section, we present the developed robotic prototype and then conduct several experiments to evaluate its performance.

### A. Developed Prototype

### B. Control with Foot Gestures

### C. Control with Head Motions

### D. Image-based Instrument Tracking

### E. Ex-vivo Cadaver Test

## V. Conclusions

## Acknowledgements

We would like to express our gratitude to Dr. Zhifeng Wang for his early mechanical design of the system.

## Appendix A
## Stability Analysis of the Jacobian Estimator

## References

[1] P. Li, H. M. Yip, D. Navarro-Alarcon, Y.-H. Liu, M. Tong, and I. Leung, "Development of a robotic endoscope holder for nasal surgery," in *Proc. IEEE Int. Conf. Information and Automation*, 2013, pp. 1194–1199.
[2] W. Lin, D. Navarro-Alarcon, P. Li, Z. Wang, H. M. Yip, Y.-H. Liu, and M. Tong, "Modeling, design and control of an endoscope manipulator for FESS," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (under review)*, 2015, pp. 1–6.
[3] H. Makino, "Assembly robot," July 27 1982, US Patent 4,341,502.
[4] P. Gerum, "Xenomai - Implementing a RTOS emulation framework on GNU/Linux," White Paper, Xenomai, Apr. 2004.