Robotic Anesthesia: a Novel Pharmacological Robot for General Anesthesia and Robot-assisted Intubation

Mohamad Wehbe

Doctor of Philosophy

Division of Experimental Surgery
Department of Surgery
Faculty of Medicine

McGill University

Montreal,Quebec

May, 2014

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Doctor of Philosophy, Experimental Surgery

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere gratitude to my supervisor, Professor Thomas Hemmerling, for his continued support and guidance throughout the period of my thesis.

I would like to thank the members of the jury for their valuable time; Professor Richard Robinson for accepting to be the internal reviewer, and Professor Maxime Cannesson for accepting to be the external reviewer of my thesis.

I would also like to thank my fellow labmates of the ITAG research laboratory for their support and technical advice as well as providing me with a great environment to work in. These thanks go in particular to Joshua Morse, Christophe Philippona, Marilù Giancalone, and the alumni: Cedrick Zaouter, Riccardo Taddei, Nora Terrasini, Erik Arbeid and Samer Charabati.

Finally, I would like to dedicate this thesis to my friends and family, without whom none of this would have been possible.

ABSTRACT

Robotic anesthesia has to deal with pharmacological patient management as well as manual gestures usually performed by anesthesiologists. General anesthesia is composed of three components: hypnosis, analgesia, and neuromuscular blockade. Different drugs are used to control each of these components. Usually, administration of these drugs is under the manual control of anesthesiologists. In order to reduce the anesthesiologist's workload in the operating room, we propose automated delivery of general anesthesia.

We have developed a closed-loop system for automated delivery of all drugs used for general anesthesia. This system delivers propofol for hypnosis control, remifentanil for control of analgesia, and rocuronium for neuromuscular blockade. The system is composed of a computer connecting the closed-loop algorithms, vital signs monitoring parameters, and three infusion pumps. The infusion rates and doses of the drugs are determined based on the feedback from the monitoring parameters.

This thesis presents a review of robotic developments in anesthesia. It presents the technological development of the closed loop system, the algorithms, the user interface and its clinical use.

The thesis also presents a system, which allows robot-assisted intubation. The technical features of this system are described as well as its testing in simulation on a standard airway mannequin and difficult airway mannequin. We hypothesized that robot assistance results in a faster skill acquisition than manual procedures for

endotracheal intubations. We used two airway mannequins, a standard airway mannequin, and an adjustable airway mannequin on two settings: standard and difficult. For every setup, 20 intubation trials were performed; intubation times were measured for each trial. Results show that robot-assisted intubations allow faster skill acquisition over manual intubation.

ABRÉGÉ

L'anesthésie robotique s'occupe de la gestion pharmacologique des patients ainsi que des tâches manuelles habituellement effectuées par les anesthésistes.
L'anesthésie générale comprend trois éléments : l'hypnose, l'analgésie et la relaxation musculaire. Différents médicaments sont utilisés pour contrôler chacun de ces composants. Habituellement, ces médicaments sont administrés manuellement par les anesthésistes. Afin de réduire la charge de travail de l'anesthésiste en salle d'opération, nous avons considéré l'administration automatique de l'anesthésie générale. À cet égard, nous avons développé un système en boucle fermée pour effectuer la livraison automatique des médicaments d'anesthésie générale. Ce système administre le propofol pour le contrôle de l'hypnose, le remifentanil pour le contrôle de l'analgésie, et le rocuronium pour la relaxation musculaire. Ce système comprend un ordinateur pour exécuter l'algorithme à boucle fermée, un moniteur d'anesthésie, et trois pompes à perfusion. Le système contrôle le débit de perfusion de façon à ce que les valeurs des signes vitaux se rapprochent le plus possible de leurs valeurs cibles.
Cette dissertation présente une revue des développements robotiques en anesthésie. De plus, elle présente le développement technologique du système en boucle fermée, les algorithmes, l'interface utilisateur et son utilisation clinique.
Cette dissertation traite également de l'intubation assistée par robot. Nous avons comparé les performances du système d'intubation sur un mannequin de voies respiratoires standard et un mannequin de voies respiratoires à difficulté ajustable. Nous

avons utilisé deux mannequins de voies respiratoires, un mannequin de voies respiratoires standard, et un mannequin de voies respiratoires à difficulté réglable sur deux réglages : standard et difficile. Pour chaque configuration, vingt essais d'intubation ont été faits. Les résultats montrent que les intubations assistées par robot permettent une acquisition de compétence plus rapide que les intubations manuelles

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

The evolution of the speciality of anesthesia began in the mid-nineteenth century and only became firmly established less than 60 years ago (Morgan et al., 2006). Ancient civilizations used poppy extract to reduce pain in order to allow surgeons to operate. It wasn't until 1846 that the field of anesthesia was revolutionized, when Dr. William Morton proved that if inhaled in proper doses, ether provides safe and effective anesthesia (Robinson & Toledo, 2012).

The evolution of anesthesia started with inhalation anesthesia, followed by local and regional anesthesia, and, finally, intravenous anesthesia (Morgan et al., 2006); however, for the purpose of this thesis, I will only introduce intravenous anesthesia in this chapter.

Intravenous anesthesia followed the invention of hypodermic needles in 1855 (Butterworth et al., 2013). The introduction of thiopental in 1934 marked the birth of modern anesthesia. Today, intravenous anesthesia is used for both the induction and maintenance of anesthesia, as well as the provision of conscious sedation (Miller, 2009). Propofol is currently the most popular agent for intravenous induction worldwide (Butterworth et al., 2013). The ideal intravenous anesthetic drug would provide hypnosis, analgesia, and muscle relaxation without affecting cardiac and respiratory functions. Since no single drug is ideal, many intravenous drugs are used in conjunction to offer the desired effects (Miller, 2009).

Researchers have been trying to improve the quality of drug delivery in anesthesia (Struys et al., 2001), and over the last two decades, closed-loop systems have been developed and used in projects in order to evaluate their advantages and disadvantages versus manual anesthesia administration (Wehbe et al., 2013).

The objectives of this project is to develop a closed-loop system for the automated delivery of general anesthesia, and to test a robot-assisted intubation system in a difficult intubation airway mannequin.

The Hypotheses of this thesis are:

- Closed-loop administration of general anesthesia is feasible and allows for a better control of the anesthesia parameters than manual control
- Robot-assistance in endotracheal intubation provides a faster skill acquisition then manual procedures

The subject of this PhD thesis is robotic anesthesia, comprising both pharmacological robots and robotic assistance for endotracheal intubation, a common procedure used to allow for artificial ventilation for patients undergoing general anesthesia.

In chapter 1, I will introduce some notions about general anesthesia, endotracheal intubation, and robotics in anesthesia.

Chapter 2 deals with the literature review, where I will also present the article entitled "Robots in Anesthesia: A Review", which was submitted to the British Journal of Anaesthesia.

In chapter 3, I will describe the technological development of the systems used in this project. In chapter 4, I will present the article entitled "A technical description of a novel pharmacological anesthesia robot" published in the Journal of Clinical

Monitoring and Computing. Details of development of the pharmacological robot, and the different modules constituting it will be discussed in this chapter.

Chapter 5 talks about the robot-assisted endotracheal intubation, the Kepler Intubation Robot (KIS). In this chapter I will present the article "Robot-assisted endotracheal intubations versus endotracheal intubations using a videolaryngoscope: comparison of learning curves using a novel difficult airway mannequin", which was submitted to the British Journal of Anaesthesia.

In the last chapter I will conclude and present future directions for this project.

CHAPTER 1
Background

1.1 Introduction

This chapter presents the different components of general anesthesia, endotracheal intubation, as well an introduction to theory that the developed system was based on.

This chapter is divided into three sections.

1.2 General anesthesia

General anesthesia began with inhaled agents but now can be induced and maintained with drugs that enter the patient through many different methods (Butterworth et al., 2013).

Induction is the administration of anesthetic drugs to induce general anesthesia and a state of unconsciousness (Chu & Fuller, 2012). Induction of general anesthesia in adults usually includes the administration of intravenous drugs (Butterworth et al., 2013).

Traditionally, general anesthesia is composed of three components: hypnosis, analgesia, and neuromuscular blockade (i.e., muscle relaxation).

Hypnosis is the loss of conciousness, it includes the blocking of sensory perception. Analgesia is the control of nociception and the inhibition of the autonomous nervous system. Finally, neuromuscular blockade is responsible for muscle relaxation to limit the involuntary reflexes provoked by surgical stimuli.

Each of these components are associated with different pharmacological agents. Thus, the hypnotic agents induce the loss of consciousness (Bischoff et al., 2008). As for pain inhibition, it is achieved through the use of opioids, while neuromuscular blockade is induced through the infusion of muscle relaxants.

1.3 Endotracheal intubation

Endotracheal intubation is the placement of a tube into the trachea in order to open the airway in order to deliver oxygen or volatile anesthetic agents. Before surgery, this is done under deep sedation and muscle relaxation.

The intubation procedure does not come without complications. If the endotracheal tube was inadvertently placed in the oesophagus, ventilation will not occur. If undetected, this can lead to brain damage and cardiac arrest. Placing the tube too deep can result in ventilating only one lung, which can in turn result in a pneumothorax as well as inadequate ventilation (Schiffman, 2014). During the procedure of endotracheal intubation, damage can also occur to the teeth, soft tissues of the mouth, and vocal cords.

1.4 Closed-loop systems

A closed-loop system is a system which provides feedback of its actual state and compares it to a desired state in order to adjust the system (VMware, Inc., 2013). In other words, closed-loop systems are able to correct their output in order to meet target results. A sensor is used to look at the output and adjust the process accordingly. Figure 1–1 shows a generic closed-loop control system.

A closed-loop system is used to replace a manual task by performing it automatically. A common example is a heating system in which a thermostat is used to measure

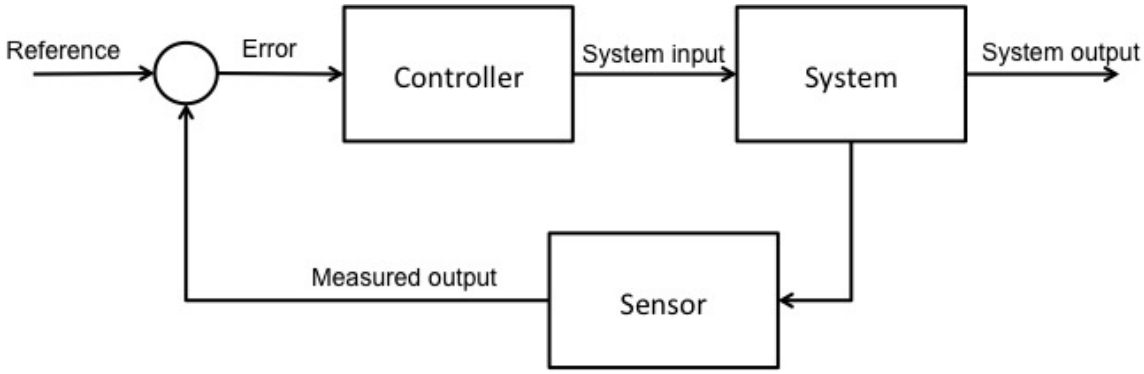


Figure 1–1: Generic closed-loop system diagram

the temperature of a room and compare it to a set target. If the room's temperature became too cold, it would turn on the heater, and if the room's temperature became too hot, it would turn the heater off. The effectiveness of this system is analyzed by monitoring how good the controller is at keeping the temperature close to the target value.

Another common example of closed-loop controllers is the cruise-control system found in automobiles.

CHAPTER 2
Literature Review

2.1 Introduction

Following the discussion of the general anesthesia and endotracheal intubation background, in this chapter we will discuss the state of the art development in the field of automated closed-loop anesthesia and robot-assisted anesthesia related procedures.

2.2 Article 1

Robots in Anesthesia: A Review

Mohamad Wehbe* and Thomas M. Hemmmerling†#

*Department of Experimental Surgery, McGill University, Montreal, Canada

†Department of Anesthesia, McGill University, Montreal, QC, Canada

#Arnold and Blema Steinberg Simulation Centre, McGill University, Montreal, QC, Canada

Contribution of Authors

- Mohamad Wehbe: idea, first draft of contents, literature review, critical analysis of literature, writing of manuscript.
- Thomas Hemmerling: supervision of contents and revision of manuscript.

Summary

Purpose of review: Robots in anesthesia are used as a tool to automate anesthesia care, reducing the anesthesiologist's workload and improving patient care. The purpose of this review is to show the latest findings in robotic anesthesia.

Recent findings: The literature separates robots in anesthesia into two groups: pharmacological robots and manual robots. Pharmacological robots are mainly closed-loop systems that help in the titration of anesthetic drugs to patients undergoing surgery. Manual robots are mechanical robots that are used to support or replace manual gestures performed by anesthesiologists. In the last decade, researchers have focused on the development of decision support systems and closed-loop systems. This has also led to interest in teleanesthesia offering the possibility of remote preoperative assessment and remote anesthesia delivery for patients undergoing surgery in remote locations.

Summary: Robots can improve performance and safety in anesthesia. In this review we present the developments made in robotic and automated anesthesia and discuss the current state of research in this field.

Keywords: Robotic anesthesia, closed-loop systems, robot-assisted intubation, robot-assisted nerve block.

2.2.1 Introduction

Anesthesia is a drug induced loss of consciousness (American Society of Anesthesiologists Task Force on Intraoperative Awareness, 2006) during which patients will require assistance in maintaining the respiratory function. Cardiovascular function might also be impaired. Consequently, a good drug control is needed in order to

perform and maintain a safe anesthesia. A well-designed automated control system can provide stability and accuracy of the controlled variables (Rinehart et al., 2012). The notion of robots in medicine is not new, and the history of surgical robots began in 1985 with the Unimation Puma 200 robot, used to perform CT-guided brain tumor biopsies (Kwoh et al., 1988). In April 1997, the first robot-assisted surgical procedure on a patient was performed in Brussels, Belgium (Himpens et al., 1998; Satava, 2002). In anesthesia however, development was slower, and the first attempt in automation was the introduction of computerized pharmacokinetic model-driven continuous infusion pumps (Schwilden, 1981; Miller, 2009). These attempts resulted in the first target-controlled infusion (TCI) device for administering propofol.

Hypnosis, analgesia and neuromuscular blockade (NMB) are the three components of general anesthesia (Miller, 2009). Monitoring the vital signs controlling these components is very important in anesthesia. The introduction of depth of consciousness monitors, such as the bispectral index (BIS) or entropy monitoring, paved the way for closed-loop anesthesia systems (Hemmerling, 2009).

More recently, research has focused on closed-loop systems in anesthesia. In contrast to TCI, closed-loop systems control the drug dose (effect) by continuously checking the controlling parameter, such as the depth of consciousness for propofol infusion. Although their performance was tested clinically, and showed to be on par with manual anesthesia, closed-loop systems are still used for research purposes and have not yet been used in clinical practice (Hemmerling, 2009).

Decision support systems have been increasingly used in anesthesia. These systems are software packages that are designed to assist trained and untrained personnel in

making correct decisions. Research has shown that decision support systems increase anesthesiologists' vigilance and patient safety during surgery (Zaouter et al., 2013; de Graaf et al., 1997). They have also been successfully implemented in clinical practices to offer suggestions about treatment to clinicians and technicians (Zaouter et al., 2013).

In this article, we review the current state of automation in anesthesia, and discuss the challenges facing it. We will discuss decision support systems, closed-loop drug delivery systems, teleanesthesia, and robot-assistance in anesthesia.

2.2.2 Decision support systems

Anesthesiologists have to monitor many patient related data during surgery, and in order to make maximum use of this information, they would have to continuously process a substantial amount of data in order to make adequate decisions (de Graaf et al., 1997). Since humans have limited processing capabilities compared to machines, help in this area is important in order to reduce human error in anesthesia. According to a study done by Halford et al., four variables is at the limit of human processing capacity (Halford et al., 2005).

Decision support systems are designed in order to reliably process the available data and present anesthesiologists with alarms accompanied by information relevant for diagnosis (de Graaf et al., 1997).

Westenskow *et al.* have worked on an intelligent alarm system to reduce the anesthesiologists response time to critical events (Westenskow et al., 1992). This system provides automatic diagnosis of critical events in the patient's respiratory system by

collecting data from respiratory sensors and diagnosing problems. This study measured human response time, between the sounding of an alarm and the resolution of fault, when using both a conventional anesthesia system and intelligent alarm systems. They simulated the patient using a test lung connected to a computer-controlled pneumatic actuator that generated random respiratory events. Their findings showed that human response times to the alarms were significantly faster when the intelligent alarm system was used.

In 1997, de Graaf *et al.* (de Graaf et al., 1997) presented a decision support system that provides only information, and not data, to anesthesiologists to assist in the assessment of the current state of patients. Their approach consisted of three phases: first, they validated the incoming data to ensure reliable processing for both the decision support system and the anesthesiologists; second, the validated data was analyzed to detect for patterns that could trigger the anesthesiologist's response; and finally, a strategic selection was made from these patterns and presented to the anesthesiologist as information relevant to the decision to be made. This fuzzy logic based system is capable of learning from each new example. This DSS was validated on the ECG of 8 patients under surgery. For every record, the abnormal periods in the first 10 minutes were manually labeled. Next, the algorithm was applied to the same records and the results were compared with the labels. Using a learning rate of 10%, 86% of all abnormal periods were classified correctly.

More recently, Zaouter et al. presented a decision support system for the management of hemodynamic and respiratory events in orthopedic patients under propofol

sedation (Zaouter et al., 2013)* . They integrated the decision support system in a closed-loop sedation system. Their decision support system is knowledge driven, collecting data from multiple signals and their reaction to medication. The described decision support system integrates alarms for low heart rate, low respiratory rates, low mean arterial pressure and low peripheral oxygen saturation. An example of these alarms is shown in Figure 2–1. These alarms are accompanied with possible reasons for their occurrence and possible options to solve the problem. They tested the system in a clinical trial with 150 patients who were randomly divided into 2 groups: one group with decision support system monitoring and one without. The reported results show that the delay to detect and treat critical events was significantly shorter in the group using the decision support system compared to the group receiving standard anesthesia, without notification or oversight with the decision support system (Zaouter et al., 2013).

Nair *et al.* presented a real-time decision support system for anesthesia care during surgery called Smart Anesthesia Manager (SAM) (Nair et al., 2013). This system works in conjunction with the Anesthesia Information Management Systems (AIMS) to provide clinical decision support. The AIMS system is useful for accurate documentation; however, it does not implement logic to detect complications during surgery. The Smart Anesthesia Manager is a notification system that can detect issues and alert anesthesiologists to take corrective measures in real time. They designed decision rules for improving quality of care, patient safety, billing and waste

* This study describes a new system for anesthesiologists that integrates automated sedation and decision support for critical events.

reduction (Nair et al., 2013). SAM was first tested in an AIMS test environment prior to migrating it to a production environment. They reported improvements in patient clinical care and anesthesia billing.

The work done in this area is mainly focused on increasing patient safety and improving clinical care and practices by making important perioperative patient data available at the clinicians' fingertips. This reduces the time to detect critical events and to resolve them.

In addition, systems like the one presented by Westenskow *et al.* (Westenskow et al., 1992) and Zaouter *et al.* (Zaouter et al., 2013) provide alarms for detected critical events along with diagnosis and possible solutions reducing the response time of clinicians.

2.2.3 Closed-loop systems

A closed-loop system is a feedback control system that monitors one or more input variables, the output signal (or a function of the output signal) and reduces the error (difference between the input and output) to bring the output of the system to a desired value (Ogata, 2010). A block diagram of a closed-loop system is illustrated in Figure 2–2.

Puri *et al.* (G. Puri et al., 2007) presented in 2007 a closed-loop anesthesia delivery system (CLADS) for the automated delivery of propofol. The presented system uses BIS as the control variable and is modelled according to the clinical pharmacokinetic and pharmacodynamic profile of propofol. A standard infusion pump served as the control actuator. The CLADS is used to control propofol infusion in order to achieve a preset BIS target both during induction and maintenance. The user

enters the target BIS value and patients' risk status. The risk status determines the maximum allowable propofol infusion rate (G. Puri et al., 2007). The system uses proportional-integral-derivative control to make changes to the propofol infusion rate based on its effect on the BIS in the previous iteration. They evaluated the system in clinical practice based on its effectiveness in maintaining a target BIS with minimal fluctuations, the economy of propofol consumption and recovery times. The authors demonstrated that the CLADS outperformed manual delivery of propofol. The BIS was maintained within a narrow range of the target with minimal fluctuations. Also with the CLADS propofol usage was much more economical and periods of excessive anesthetic depth were lower.

In 2011, Liu *et al.* presented a closed-loop controller that co-administers propofol and remifentanil guided by the bispectral index (Liu et al., 2011)* . They developed the controller using a proportional-integral-derivative algorithm. This controller guides the IV co-administration of propofol and remifentanil during induction and maintenance of general anesthesia, by BIS monitoring. They tested the system on 167 patients randomly assigned to dual closed-loop (83 patients) or manual (84 patients) TCI propofol and remifentanil. Their platform consisted of software implemented on a personal computer that served: to calculate the effect-site concentration of propofol and remifentanil and display it on the screen, as a user interface to enter patients' demographic data and to modify the concentrations, to control the infusion pumps for propofol and remifentanil, and to collect data every 5 seconds (Liu et al., 2011).

* This study points out the importance of integrated control of two drugs delivered through a closed-loop system.

The initial propofol dose was set by the clinician according to his/her clinical judgement and based on this the controller fixed the initial remifentanil concentration. Thereafter, modifications of propofol and remifentanil effect-site concentrations were decided by the controller based on rules. The results showed that the closed-loop system allowed for an improvement over manual control, with an increase in time spent with BIS values within predetermined boundaries (between 40 and 60), and a decrease in the median absolute performance error. The closed-loop system also allowed for a decreased induction phase and extubation times.

This system was also tested in a more recent randomized clinical trial on 67 patients during rigid bronchoscopic procedures (Liu et al., 2013). Thirty-four patients were included in the manual TCI group and 33 were included in the dual-loop group. This study however could not establish the superiority of the automatic system over manual adjustment for bronchoscopy. The percentage of time spent with BIS values within predetermined boundaries was similar for both groups. The durations of induction and maintenance and time to tracheal extubation were also similar between groups. This could be explained by the unique characteristics of anesthesia for rigid bronchoscopic procedures, especially a short duration of maintenance, and electroencephalogram artifacts caused by inadequate neuromuscular blockade (Liu et al., 2013).

Hemmerling *et al.* (Hemmerling et al., 2013)** developed a novel closed-loop total intravenous anesthesia drug delivery system called McSleepy. McSleepy is an

** This study describes a system that controls the three components of general anesthesia in a closed-loop.

automated closed-loop anesthesia drug delivery system that integrates all three components of general anesthesia. They used BIS as the control variable for hypnosis (propofol infusion), a pain score derived from heart rate and mean arterial pressure (Analgoscore (Hemmerling et al., 2009)) as the control variable for analgesia (remifentanil titration), and phonomyography (Relaxofon (Wehbe et al., 2012)) to control neuromuscular blockade. McSleepy was designed to perform all three stages of general anesthesia: induction, maintenance and emergence. The maintenance interface is shown in Figure 2–3. Initial drug doses for the induction stage are preset in the system according to the anesthesiologist's clinical judgement; subsequent doses are defined by the system following the algorithm and based on the patient's demographic data and vital signs. This system was tested in a randomized clinical trial on 186 patients divided into 2 groups of 93 patients each. The authors tested the system for effectiveness in maintaining target BIS and Analgoscore values (Hemmerling et al., 2013). Results showed better control of hypnosis and nociception with shorter periods of over- or undershoot and faster extubation times than manually administered anesthesia.

The presented closed-loop systems are developed in order to improve precision of drug dosage based on the patient's vital signs. These systems are able to perform more drug dose modification per hour compared to anesthesiologists due to the fact that they are constantly monitoring the vital signs, resulting in lower drug consumption and better patient recovery.

While all presented systems are used to deliver general anesthesia, the CLADS is used to control only component of general anesthesia, hypnosis, and the system presented

by Liu *et al.* controls 2 components of general anesthesia, hypnosis and analgesia. Only McSleepy is developed to control all 3 components of general anesthesia.

2.2.4 Closed-loop systems for sedation

Closed-loop systems were also developed to perform sedation. The SEDASYS System (Ethicon Endo-Surgery, Cincinnati, Ohio) is a computer-assisted personalized sedation system (Pambianco et al., 2008, 2011). The SEDASYS (Figure 2–4) integrates the monitoring of pulse oximetry, capnometry, ECG, noninvasive blood pressure, and patient responsiveness, with the delivery of oxygen and propofol (Pambianco et al., 2008). The SEDASYS is designed to facilitate the safe administration of propofol to induce minimal to moderate sedation to relatively healthy adults undergoing elective colonoscopy or esophagogastroduodenoscopy (EGD) (Banerjee et al., 2011). The SEDASYS system is operated by an endoscopist-nurse team whose members are trained in general anesthesia (Banerjee et al., 2011). The device also incorporates an automated responsiveness monitor (ARM) to assess patient responsiveness. The ARM delivers at preset time intervals, an auditory request asking the patient to squeeze the handset he/she is holding, together with a mild vibration. If the patient fails to respond by squeezing the handset, the auditory request becomes louder and the vibration more vigorous. The ARM calculates the patient's response time (Banerjee et al., 2011). The SEDASYS will not allow propofol infusion unless oxygen is being delivered to the patient. Signs of oversedation trigger the system to increase the oxygen delivery rate. Also, the system automatically decreases the propofol rate when responsiveness to the ARM is lost (Banerjee et al., 2011). In a study published in 2011 by Pambianco et al., the SEDASYS was used to perform

sedation on 489 patients undergoing routine EGD and colonoscopy (Pambianco et al., 2011). The 493 patients of the control group received traditional sedation with benzodiazepine and opiods. The study proved that the SEDASYS reduces risks of oversedation, and patients receiving sedation with the SEDASYS experienced fewer and less significant oxygen desaturation than patients in the control group.

On the 3rd of May 2013, Sedasys announced that the U.S. Food and Drug Administration has granted premarket approval for the SEDASYS system (Johnson, 2013). The SEDASYS system is expected to be introduced on a limited basis beginning in 2014.

2.2.5 Teleanesthesia

Telemedicine is the delivery of healthcare and sharing of medical knowledge over a distance using telecommunication systems (Strode et al., 1999). It has been used to overcome the shortage of qualified specialists in remote areas (Strode et al., 1999; Cermack, 2006; Galvez & Rehman, 2011). Although telemedicine has been used by surgical specialities, the evaluation of telemedicine for anesthesia is fairly new.

In 2004, Wong *et al.* (Wong et al., 2004) developed a system for remote preoperative assessment and tested it in a pilot study. The aim of the study was to provide telemedicine clinical consultations to residents of central and northern Ontario in Canada (Wong et al., 2004). They equipped both sites (local and remote) with videoconference monitors and cameras to allow live two-way communication. At the consultant site, there was an anesthesiologist and at the remote site a nurse helped the patient during the consultation. The anesthesiologist took a history from the patient. Mouth opening and the Mallampati score were assessed using the airway

camera. The airway profile, thyromental distance and neck movement were assessed using the room camera. A digital stethoscope was used auscultate the heart and lung sounds. This study required patients to travel to a specific location equipped with specialized equipment, a dedicated network between local and remote locations, and trained medical personnel. This program is now part of the Northern Ontario Remote Telecommunication Health (NORTH) Network, based in Toronto, and provides services to over 65 sites throughout Ontario and Manitoba.

Other studies focused on remote anesthesia monitoring (Cone et al., 2004; Fiadjoe et al., 2009). Cone *et al.* (Cone et al., 2004) reported a case where they used telemedicine and telemonitoring technologies during anesthesia and general surgery between a remote sector of Ecuador and their facility at Virginia Commonwealth University. Patient monitoring and data transmission with audio and video communication capabilities was conducted through a "rapidly deployable telemedicine unit", which allowed the integration of physiologic data, including ECG waveforms, oxygen saturation, arterial blood pressure, breath sounds, voice contact, and field video capture. They used a satellite Internet connection with a bandwidth of 64 Kbps for real-time transmission of the data throughout the procedure. In the case they reported, the effectiveness of the application was demonstrated by collaborative intervention to correct a period of premature ventricular contractions progressing to ventricular bigeminy (Cone et al., 2004).

More recently, Hemmerling et al. (Hemmerling et al., 2013)** presented a pilot study

** This article describes the clinical application of the anesthesia system McSleepy in the world's first transcontinental anesthesia.

where they performed transcontinental anesthesia between the Montreal General Hospital, Montreal, Canada and the Cisanello Hospital, Pisa, Italy. A diagram showing the set-up and operation is shown in Figure 2–5. At the patient's site (in Pisa) one computer was set-up with an automated anesthesia delivery program (McSleepy) to control delivery of anesthetic drugs, and a second computer was set-up with live feeds of four webcams for different monitoring purposes: the automated anesthesia delivery system interface, a view of the surgical field, a view of the GlideScope during intubation, and a view of the vital signs monitor. At the anesthesiologist's site (in Montreal), one computer was used to control the automated anesthesia delivery program using remote desktop control software TeamViewer (TeamViewer Version 6, Goppingen, Germany), a second computer was connected with the video-feed computer in Pisa via Skype. Both local computer systems and remote computer systems were connected via a standard Internet connection with a high bandwidth of up to 8 Mbps. Transcontinental anesthesia was performed on 20 patients undergoing thyroid surgery in Pisa. They defined the success of transcontinental anesthesia as induction, maintenance and emergence from anesthesia without necessitating intervention of the local anesthetic team. The study showed that the clinical performance was good, and that transcontinental control of an automated anesthesia delivery system is possible via a standard Internet connection.

2.2.6 Robotic assistance in anesthesia

Robots are reliable, never fatigue, extremely precise with near-absolute geometric accuracy (Louw et al., 2004). These advantages over manual human dexterity underline the development of robotic assistance in anesthesia. Research into robotic

assistance not only offers the physical advantages described above, but also provides the possibility of conducting anesthetic procedures remotely, a requirement for any fully-functional teleanesthesia system.

Tighe *et al.* presented a robotically assisted fiber-optic intubation on an adult airway mannequin (Tighe, Badiyan, Luria, Lampotang, & Parekattil, 2010). They used a multi-purpose da Vinci Surgical System type S (Intuitive Surgical, Sunnyvale, California) to perform an oral and nasal intubation. The da Vinci system has four arms; the first arm, equipped with a camera, was positioned above the mannequin. The second and third arms were equipped with large and small graspers to simulate hand movement. The fourth arm was equipped with a standard fiber-optic bronchoscope (Karl Storz Endoscopy, El Segundo, California). A da-Vinci-Surgical-System-trained urologist performed the two intubations with manual assistance from an anesthesiologist in certain steps of the procedures. They reported successful completion of the two intubation attempts. They had considerable difficulty using the robotic graspers to lift and manipulate the laryngoscope handle.

At the same time, a robotic intubation system called the Kepler Intubation System (KIS) was being developed at McGill University in Canada (Hemmerling et al., 2012)** . They developed a robotic system to perform tracheal intubation using a standard video laryngoscope. This system is shown in Figure 2–6. The system consisted of a robotic arm controlled by a joystick. The robotic arm is capable of performing simulated wrist and arm movement which allows it to perform intubation

** This study describes the first robotic tracheal intubation system tested on humans.

similar to human anesthesiologists (Hemmerling et al., 2012). The KIS is composed of four main components: a joystick, a robotic arm, a video laryngoscope, and control software. They tested their system on 12 patients, and a trained anesthesiologist performed all the intubations. The KIS user interface receives two camera feeds, one from the video laryngoscope, and the other from a camera positioned laterally to the patient (Hemmerling et al., 2012). They reported a success rate of 91% (11 out of 12 patients), where in one patient the KIS intubation was aborted due fogging of the laryngoscope video.

Another area where robotic assistance is being developed is regional anesthesia. In a similar fashion to what was mentioned above, Tighe *et al.* used the da Vinci Surgical System to perform a robotically-assisted ultrasound-guided nerve block (Tighe, Badiyan, Luria, Boezaart, & Parekattil, 2010). This was done in a simulated environment as a single-injection nerve block, and placement of a perineural catheter into an ultrasound phantom under ultrasound guidance. As with their other study mentioned above, they used a da Vinci Surgical System to perform the procedure. They tested first the ability of the robotic system to manipulate the equipment involved in a peripheral nerve block. After this test, they placed the ultrasound phantom on an operating room stretcher under the da Vinci system. The manually placed the ultrasound probe, and after identifying the simulated perineural structures in the phantom, they stabilized the probe with the da Vinci system. Three arms were used to maneuver most of the equipment relevant to the nerve block. Their study proved that robotically-assisted regional anesthesia is feasible, however the use of a multimillion dollar system to perform regional anesthesia, along with the number

of personnel needed to accomplish this task (2 engineers, an anesthesiologist and a urologist) is not practical.

Another system designed to perform robot-assisted, ultrasound-guided nerve blocks is the Magellan system, designed and developed at McGill University in Canada (Morse et al., 2013). The Magellan has 4 components: a standard nerve block needle and syringe mounted via a custom clamp to a robotic arm, an ultrasound machine, a joystick, and control software. This system, shown in Figure 2–7, was designed to work with any ultrasound machine with a video output. The ultrasound video output is captured and displayed on the user interface of the control software. This system was tested on 13 patients undergoing surgery below the knee with sciatic nerve block anesthesia (Hemmerling et al., 2013a)** . Nerve identification was performed manually using a standard ultrasound probe. Once the sciatic nerve was identified, the probe was held in position and the Magellan was operated by a trained anesthesiologist. Among the 13 patients enrolled in the study, 3 underwent bilateral lower limb surgery, making a total of 16 blocks. All 16 robot-assisted nerve blocks were successful. They had a local anesthetic spread around the nerve sheath, directly visualized on the ultrasound video (Hemmerling et al., 2013a).

2.2.7 Automation and patient safety

In the 2000 report, *To Err Is Human* (Kohn et al., 2000), the Institute of Medicine estimated that between 44,000 and 98,000 Americans die each year as a

** This article describes the clinical application of a robot-assisted nerve block systems performed on humans.

result of medical errors. Errors are viewed as unsafe acts arising primarily from aberrant mental process such as forgetfulness, inattention, poor motivation, carelessness, negligence, and recklessness (Reason, 2000). The main source of human error in anesthesia is the high number of variables an anesthesiologist has to monitor: up to 100 parameters, while the human brain cannot simultaneously process more than 4 or 5 parameters (Halford et al., 2005). Automation and decision support systems could be used to reduce medical errors. Automated systems provide increased precision and faster computation compared to humans, they also do not fatigue. Decision support systems reduce the amount of data clinicians have to monitor; they notify clinicians when critical events occur with treatment options, allowing them to be alert at all time during surgery. According to a recent review analyzing 70 studies, clinical practices were improved by 68% of cases when decision support systems were integrated into the clinicians' work (Kawamoto et al., 2005).

2.2.8 Conclusion

Robots in anesthesia are useful operating tools that help mitigate the hazards created by the fatigue and manual accuracy of anesthesiologists. They help reduce the anesthesiologists' workload by monitoring vital signs, detecting critical events and notifying the attending clinician.

Robots in anesthesia can be separated into two types: pharmacological robots and manual robots. Pharmacological robots are mainly closed-loop systems that help to administer anesthetic drugs, while manual robots are robots that are used to support or replace a manual gesture performed by anesthesiologists.

On top of the safety and accuracy that robots add to anesthesia, they also contribute

to development and advancement of teleanesthesia. As described above, researchers are using robots to perform teleanesthesia.

There are several challenges facing further implementation of robotics in anesthesia: compared to surgical robotics, there is a relatively small amount of research being done in this field. Consequently, little funding opportunities exist for these projects. Also, a wide range of regulatory, business, and clinical issues need to be resolved before anesthetic robots can be introduced into the market and incorporated into general practice (Manberg et al., 2008).

Figures

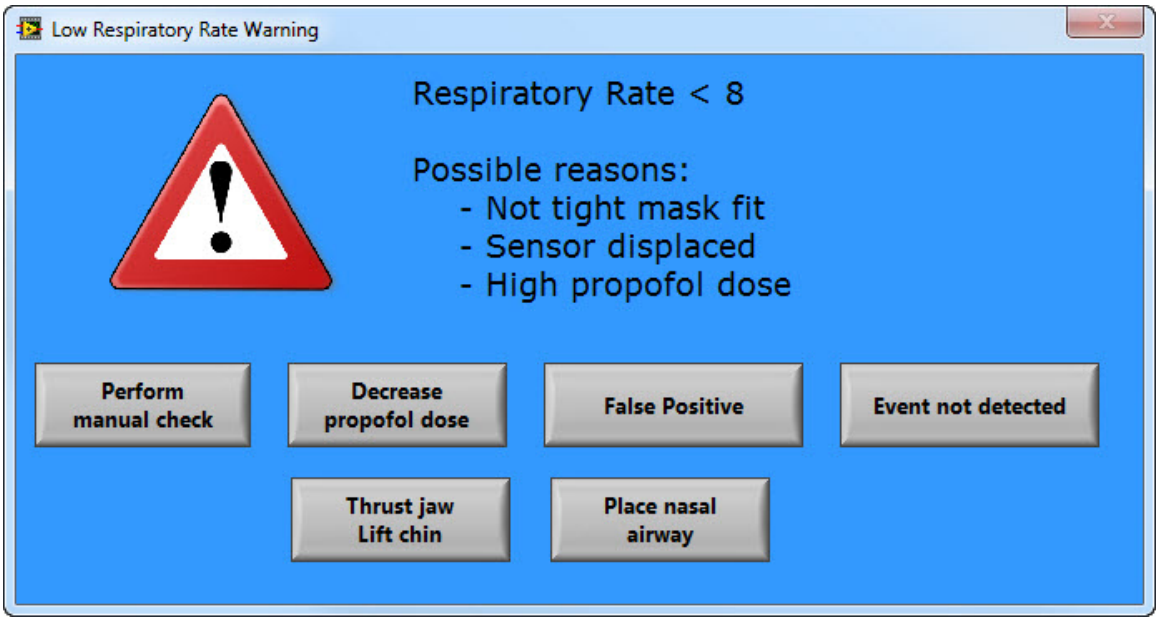


Figure 2–1: Low respiratory rate critical event from the DSS by Zaouter *et al.* The alarm displays possible reasons for the occurrence and potential solutions.

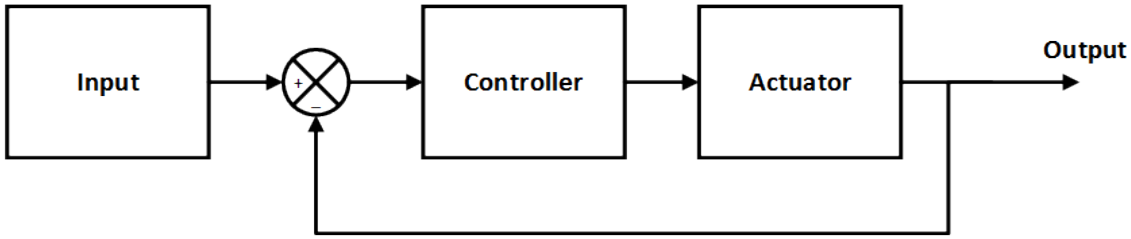


Figure 2–2: Block diagram of a simple closed-loop control system.

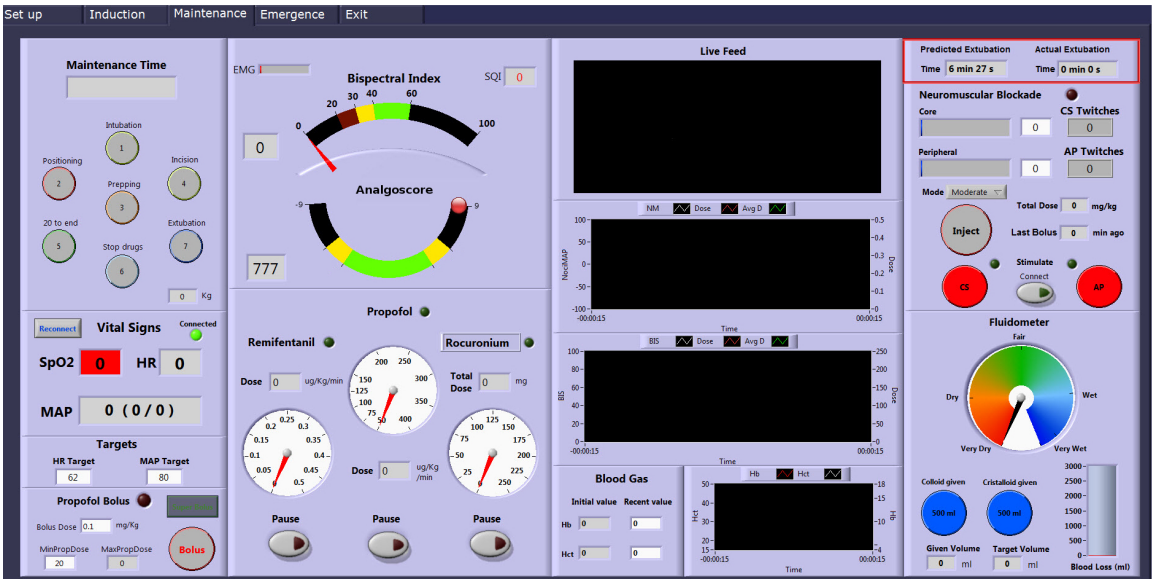


Figure 2–3: The McSleepy maintenance interface (Hemmerling *et al.*).

Figure 2–4: The SEDASYS system: bedside monitoring unit (left) and procedure room unit (right).

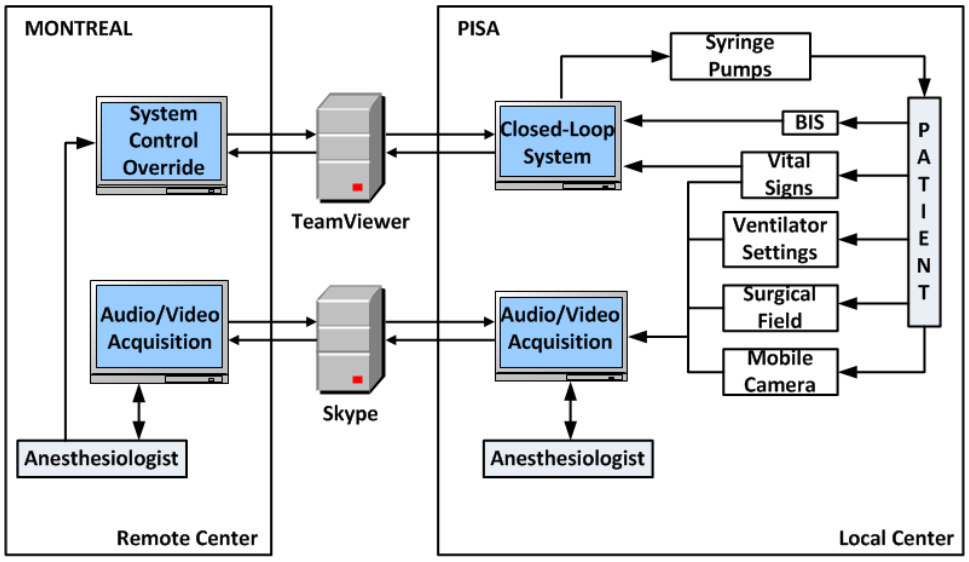


Figure 2–5: Block diagram showing the transcontinental anesthesia set-up (Hemmerling *et al.*).

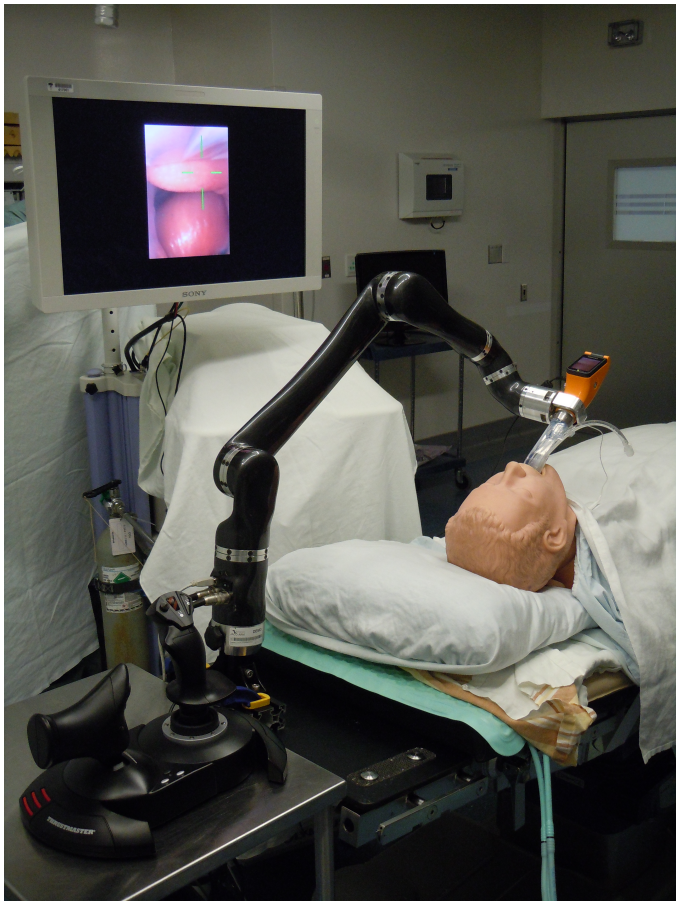


Figure 2–6: Illustration of the Kepler Intubation System. The main components of this system are shown: joystick, robotic arm, video laryngoscope and video feed.

Figure 2–7: The Magellan robotic nerve block system.

2.3 Closed-loop systems

2.3.1 Unconsciousness

2.3.1.1 Early work

One of the first closed-loop feedback control systems in anesthesia was created by Schwilden and colleagues in the late eighties (Schwilden et al., 1987, 1989), before the introduction of the commercially available depth-of-consciousness monitors.

In 1987, Schwilden *et al.* (Schwilden et al., 1987) used a combined pharmacokinetic and pharmacodynamic model to establish and evaluate feedback control of methyhexital anesthesia in 13 volunteers. They use the median electroencephalography (EEG) frequency of the power spectrum and a model-based, adaptive controller to administer the anesthesia. The controller setpoint was 2-3 Hz. The closed-loop system was able to maintain the EEG within the target area 75% of the time that anesthesia was administered.

In 1989, the same research group used a similar setup to their earlier study to administer propofol for 2 hours to 11 volunteers to a target of 2-3 Hz of median EEG frequency of the power spectrum (Schwilden et al., 1989). An average median frequency of 2.5 ($\pm$0.3) Hz was measured.

2.3.1.2 The CLAN (closed-loop anesthesia)

In 1999, Kenny *et al.* (Kenny & Mantzaridis, 1999) presented a closed-loop control system of propofol using auditory evoked potentials (AEP) as feedback. They developed a technique using AEP to extract an index to measure the depth of anesthesia. They employed a proportional integral control algorithm using a 3 s running average of the AEP index to control a target-controlled infusion pump to deliver

propofol. They induced analgesia in 100 patients using target-controlled infusion of alfentanil at a plasma concentration of 15 ng/ml. Patients undergoing this study did not receive neuromuscular blockade agents, and ventilation was mostly spontaneous breathing. The controller performance was very good, with the AEP maintained within the control value ± 5 for an average of 65.2% of the duration of anesthesia. The Varvel performance parameters were not presented in this study.

In 2002, they followed up with another study using the same type of general anesthesia (spontaneous breathing) in 10 patients undergoing major orthopedic surgery, using bispectral index (BIS) as the feedback control parameter (Absalom et al., 2002). Analgesia in this study was achieved using epidural analgesia. Although the presented Varvel performance parameters are satisfactory, the results were significantly worse than in their previous study (Kenny & Mantzaridis, 1999). With the feedback from AEP, the measured values were within 5% and 15% of the target in 65% and 99% of time, respectively, whereas with the BIS feedback, it was only in 34% and 75% of time, respectively.

A year later, Kenny's research group tried to improve the control algorithm of their system and presented the results in a follow-up study on 20 patients undergoing body surface surgery of only 27 min on average (Absalom & Kenny, 2003). In this study, they used a newer BIS software, adjusted the gain constants used in the control algorithms, altered the control algorithm to allow effect-site steering, rather than blood site steering (only in the second part of the study: patients 11-20), in an attempt to avoid oscillations by faster increase in effect site concentrations with too light anesthesia. The results showed slightly improved Varvel performance parameters from

the earlier study; however, it is worth noting that this was a very different study population and type of surgery, it would have been more worthwhile to have tested the improved control algorithms on the same patient population, study setup, and surgery type.

2.3.1.3 Work from Struys's and De Smet's research group

Following the work done by Schwilden *et al.* (Schwilden et al., 1989), the research group of Struys's and De Smet's developed a closed-loop propofol sedation system around model-based controllers incorporating target-controlled infusion technology combined with a pharmacokinetic-dynamic model. The designed controller is adaptive, meaning the dosing is adjusted according to individual patient reactions to propofol. This adaptive system was first presented in 1998 on 10 patients (aged 18-65, ASA I and II) undergoing elective orthopedic surgery under spinal anesthesia (Mortier et al., 1998). The propofol was administered using a target-controlled infusion pump connected to a computer over a RS-232 interface. The induction was automatic via a stepwise target effect-site controlled infusion with no feedback from BIS. The propofol target was set clinically at 2 μg/ml, and once that target was reached, the BIS value was used to steer the closed loop. The article, however, lacks information about clinical performance parameters.

In a subsequent study, they compared the performance of the BIS-controlled closed-loop administration of propofol to its manual counterpart (Struys et al., 2001). In this publication they gave details about the adaptive aspect of their design. A control action is implemented into the algorithm, which uses the difference between two consecutive measured BIS values multiplied by a differential factor.

This study has some limitations. First, the small sample size of 10 patients in each group, and secondly, they used BIS at a target of 50 in the closed loop group, whereas they used clinical signs of adequate anesthesia in the control group. This limits the possibility to compare the performance of anesthesia control. The comparison of controller performance using Varvel's parameters is also less valid since BIS was not a control value in the control group.

2.3.1.4 The CLADS (closed-loop anesthesia delivery system)

The CLADS is a novel closed-loop anesthesia delivery system developed by the research group of Puri and presented in 2007 (G. Puri et al., 2007). A big difference to the systems mentioned above, the CLADS can be used both for induction and maintenance of anesthesia. As with most other systems, however, the CLADS uses the BIS as the control parameter within the feedback system, consisting of a syringe pump, propofol, and an algorithm controlling the propofol infusion. They system has a user interface allowing users to operate it manually via a computer. The user is able to select a target BIS value, and the maximum infusion rate is related to ASA classification, where ASA I-III is considered low risk, and anything above is considered high risk. The system uses a proportional integral differential, based on the current infusion rate and the change in BIS values.

In the study they published in 2007, Puri and colleagues compared results from 20 patients where hypnosis was delivered using CLADS versus results obtained from 20 patients where hypnosis was performed manually to maintain a BIS of 50. Analgesia was maintained using fentanyl (1 μg/kg/h), and neuromuscular blockade using vecuronium boluses in both groups (G. Puri et al., 2007). They reported that the

CLADS outperformed the manual group in terms of maintaining the BIS close to target (± 10) on average 87% of time versus 77% of the time in the manual group. Also, CLADS provided anesthesia using a significantly lower dose of propofol, thus creating less overshoot of the BIS.

In the first study, patients were around approximately 40 years of age, with normal body weights undergoing elective non-cardiac surgery. In a follow-up study, Puri and colleagues applied the system to a sicker patient population undergoing cardiac surgery (Agarwal et al., 2009). In this study, they used the same design and they included 37 patients randomized into 2 groups (19 and 18 patients; CLADS versus manual propofol control). In this study, they were able to confirm their earlier findings: the CLADS was able to induce anesthesia with less BIS overshoot, which resulted in better hemodynamic control.

In 2010, they used the CLADS to control postoperative sedation in the intensive care unit to maintain a BIS target of 70 (Solanki et al., 2010). In this study as well, they reported a better performance of the CLADS in maintaining BIS around the target. Recently, they showed that their system performs well in patients undergoing abdominal or orthopedic surgery in general anesthesia at high altitude (3505 m above sea level) (G. D. Puri et al., 2012).

The same group recently adapted their system for use with volatile anesthetics, called the improved anesthetic agent delivery system (IAADS) (Madhavan et al., 2011). They used the IAADS in 40 patients undergoing cardiac surgery, randomized into 2 groups of 20 patients, where in one group, patients received isoflurane through a closed-loop (IAADS), and in the second group through a Tech 7 vaporizer adjusted

manually to achieve a target BIS of 50. In both groups, patients were induced with a propofol infusion and isoflurane was started after intubation. They reported that their system was able to maintain BIS within ±10 of target for a significantly longer period of time (84.6 ± 7.2% in IAADS group vs. 75.9 ± 11.2% in manual group, $p < 0.01$).

2.3.1.5 Liu's research group

In 2006, Liu and colleagues presented a novel closed-loop system for induction (Liu, Chazot, Trillat, et al., 2006) and maintenance (Liu, Chazot, Genty, et al., 2006) of propofol. Their closed-loop system used BIS monitoring as feedback control, and the output was via target-controlled infusion of propofol. The target bispectral index was set at 50, with minimal and maximal propofol target concentrations set at 1 and 5 μg/ml; these values could be modified by the user. Analgesia was manually controlled using target-controlled infusion of remifentanil. This system was used in a preliminary study only for induction (defined as the time between the start of anesthesia and reaching a target BIS of 50, therefore before intubation) in 40 patients, randomized in two groups of 20: closed-loop induction or manual induction using target-controlled infusion (Liu, Chazot, Trillat, et al., 2006).

They reported lower consumption of propofol and faster induction time to reach the target value with the closed-loop system. No hemodynamic difference between the groups was reported.

The follow-up study was on a larger scale, where they recruited 164 patients randomized into two groups of 81 (manual control) and 83 (closed-loop control) (Liu, Chazot, Trillat, et al., 2006). In this study however, the duration of the induction

was significantly longer in the closed-loop group than in the manual group, using a significantly higher dose of propofol. Yet, the clinical performance parameters and the Varvel parameters of system control efficiency were significantly better in the closed-loop control. In 2008, Liu and colleagues showed that closed-loop anesthesia is possible even in the most difficult surgeries, by presenting a case study of 20 patients undergoing bilateral (n=14) or single (n=6) lung transplantation (Liu et al., 2008). They reported clinical control values similar to less sick patient population, where their system was able to maintain anesthesia within acceptable BIS values of 40-60 in 84% of the time, with limited overshoot and undershoot in 13% and 3% of the time, respectively. Later, Liu *et al.* followed by creating a dual-loop system to titrate both propofol and remifentanil in closed-loop control using only the BIS as feedback control for both loops. They were based on the assumption that small changes in the BIS are an indication of nociception (painful intra-operative stimuli provoke cortical activation and consequently a BIS change) and therefore only remifentanil concentrations are affected; on the other hand, larger changes in BIS are indications of changes in hypnosis and nociception, hence both drug concentrations are changed. They also implemented an interaction rule between propofol and remifentanil: if the controller successively increases the remifentanil concentration more than 3 times, then the propofol concentration is increased (Liu et al., 2011). In 2011, they tested the dual-loop system in a study of 167 patients undergoing surgery randomized in two groups: 83 patients in a group using the dual-loop propofol-remifentanil control system, and 84 in a group using manual control. As is the case with other closed-loop studies, they reported better control of BIS values

around the target. They also reported quicker extubation times in the dual loop group.

In 2012, they followed in the footsteps of extracting two components, one for pain and one for hypnosis, from the depth of consciousness monitoring parameter by using the spectral M-entropy monitor as guidance for dual-loop control system (Liu et al., 2012). In a recent study, the same research group tested their dual-loop system for deep sedation in critically ill patients (Guen et al., 2013).

2.3.1.6 The Rostock Assistant System for Anesthesia Control (RAN)

In 2009, a research group from the university of Rostock presented a multiple-input-multiple-output controller for closed-loop control of target infusion of propofol using BIS feedback, and infusion of mivacurium using EMG feedback (Simanski et al., 2009). The two control systems work independently and are based on different control strategies. The RAN uses a conventional adaptive generalized predictive controller to control the neuromuscular blockade.

In 2011, they evaluated the clinical performance of this combined system for general anesthesia, in a study on 20 patients undergoing intra-abdominal or orthopedic surgery (Janda et al., 2011). The controller performance of the propofol system was similar to other closed-loop studies, with BIS within 10% of the target during 65% of the time; the controller performance of the mivacurium system was also similar with 87% of the time within 10% of the target.

2.3.2 Closed-loop control of neuromuscular blockade

Early closed-loop systems for neuromuscular blockade control used electromyography as a feedback method; in part due to the ease of communication it provided

over RS232, a standard communication protocol. The controllers range from simple on-off controllers, to proportional-integral-derivative controllers, to adaptive model-based controllers.

Olkkola *et al.* described a closed-loop control of atracurium-induced neuromuscular blockade using a model-based adaptive feedback algorithm (Olkkola et al., 1991). They used the Relaxograph® neuromuscular transmission monitor (Datex, Helsinki, Finland) to obtain the electromyographic control values. They used an infusion pump as the actuator of the closed-loop control. In their algorithm, they used a two-compartment model with a hypotetic effect compartment linked to the central compartment to represent a valid pharmacokinetic-dynamic model.

In 1998, Edwards and colleagues presented a 'self-learning' fuzzy logic control feedback system used to administer atracurium to a required depth of neuromuscular blockade (Edwards et al., 1998). They used a Paragraph neuromuscular monitoring device to measure the degree of neuromuscular blockade and control it in a way that the first twitch of the train-of-four is kept at 10% of its baseline value. Their controller instructed an infusion pump to administer atracurium in order to maintain this level of blockade. They tested the system on 10 patients undergoing surgery expected to last longer than 90 min. The system was able to achieve stable control of neuromuscular blockade with a mean error for the first twitch of the train-of-four of -0.45 (-1.06 to 0.13)%.

2.3.3 Control of volatile anesthetics

2.3.3.1 The ZEUS: target-controlled infusion of volatile anesthetics

The ZEUS anesthesia machine (Dräger Medical, Lübeck, Germany) is a commercially available device that includes an auto-control mode, which allows targeting end-tidal volatile and inspired oxygen concentrations. The ZEUS anesthesia machine includes a blower-driven ventilator, an electronically-controlled gas and vapor delivery system and a servo-controlled valve system. It allows for either manual control of gas delivery, or an auto-control where the fresh gas flow is automatically adapted to the patient's uptake for both oxygen and volatile agents (Lortat-Jacob et al., 2009). The inspired and expired concentrations are measured by sidestream infrared technology and paramagnetic oxygen sensors. The sample gas flow is then fed back to the system to close the loop.

This system can have some clinical advantages, such as less over and undershoot when end-tidal concentrations need to be changed (Sieber et al., 2000), lower costs (Curatolo et al., 1996) and reduced workload for anesthesiologists (Lortat-Jacob et al., 2009). However, it does not adjust the end-tidal concentration of the volatile agents according to the depth of unconsciousness or other patient parameters.

2.3.3.2 Closed-loop systems for volatile anesthetics

As described above, the ZEUS is not a true closed-loop controller for volatile anesthetics. There are only a few published studies that describe the design and clinical testing of such systems, using depth of consciousness as feedback control. One such system has already been described in section 2.3.1.4, presented by Puri and colleagues (Madhavan et al., 2011).

Another closed-loop system for volatile anesthetics was described by the research group of Zbinden and Gentilini (Gentilini et al., 2001; Locher et al., 2004). The system uses isoflurane as the volatile agent. They adopted a cascaded internal model control system to control the BIS, that provides endtidal concentration reference values to the system, and an inner controller that incorporated feedback from the BIS into the model-based algorithm to maintain a target level of hypnosis.

The controller acquires the BIS every 5 s, and regulates the dose accordingly. It uses a model-derived external controller similar to those used for intravenous agents. The group compared this system to manual administration of isoflurane in a controlled trial on 20 patients randomized into two groups of 10 (Locher et al., 2004). Isoflurane was guided by BIS at a target of 50 for a mean duration of two hours. Analgesia was maintained manually in all patients. The Varvel performances favor the closed-loop control; however with no presentation of clinical performances, an assessment of the significance of this difference cannot be made.

2.3.4 Closed-loop control of analgesia

Unlike hypnosis and neuromuscular blockade, where there are monitoring parameters available, there are no monitoring parameters for assessing pain during unconsciousness. However, most anesthesiologists use hemodynamic changes as an indirect parameter to assess nociception during surgery.

One of the first attempts to use hemodynamic parameters for pain assessment clinically was published by Carregal and colleagues in 2000 (Carregal et al., 2000).

They designed a fuzzy logic controller in C language. The described controller adjusts alfentanil infusion based on changes in the mean arterial pressure and heart rate. They used this controller in eight patients (ASA I-II) undergoing gynecological surgery under anesthesia with propofol and alfentanil. The mean arterial pressure was within ±15% of the target for 92.2% of the time the controller was used. No results about the heart rate stability were given.

In 2001, Gentilini and colleagues presented a feedback controller they designed for the automatic delivery of analgesic drugs during surgery (Gentilini et al., 2001). The system controlled the infusion rate of alfentanil via a computer-controlled infusion pump. The controller regulates two outputs: the patient's mean arterial pressure and heart rate. In the design of the controller, they adopted a linear pharmacokinetic-pharmacodynamic to describe the predicted concentration of alfentanil in the plasma. They tested their model in 13 healthy patients undergoing minor surgery (Luginbuhl et al., 2006). In this study, the controller used only mean arterial pressure to titrate alfentanyl in a closed-loop fashion. They induced alfentanyl manually, and then they switched it to closed-loop control in order to maintain the mean arterial pressure at a set-point of 70 mmHg.

The mean absolute deviation of mean arterial blood pressure from target was 10 mmHg during surgery, which is equal to a mean offset of approximately 15%. The controller performances were not compared to a control group, which limits the assessment of the results.

One study presented a composite pain score called the 'Analgoscore' that uses mean arterial blood pressure and heart pressure as the basis for the score (Hemmerling et

al., 2009). The Analgoscore ranges from -9 (too profound analgesia) to +9 (insufficient analgesia), with 0 as a target of perfect balance between analgesic treatment and pain. The control mechanism is an adaptive proportional-integral-derivative controller, which analyses past offset in integral time intervals and adjusts the infusion rate accordingly.

In a controlled clinical trial, the Analgoscore was used to titrate remifentanil in 27 patients undergoing surgery. In one group of 16 patients, remifentanil was administered using a closed-loop with Analgoscore as the feedback value: the target was set at the range of -3 to +3 (considered as excellent analgesia control). In the control group of 11 patients, anesthesiologists were advised to use the Analgoscore as guidance for remifentanil infusion rate settings with the goal to maintain a value of 0 as much as possible. The closed-loop group was in the range of excellent analgesia control 84% of the time versus 70% of the time for the control group.

2.4 Manual robots

In this section, when we address manual robots, we are interested in handling devices with manual, often remote control. In anesthesia, there are two areas where manual robots have been used: intubation and regional anesthesia.

2.4.1 Robot-assisted intubation

Tighe *et al.* used the da Vinci surgical robot to assist endotracheal intubation in an airway mannequin (Tighe, Badiyan, Luria, Lampotang, & Parekattil, 2010). They performed two fiberoptic intubations in simulation. One of the da Vinci's arms was equipped with a camera and placed over the head of the mannequin. Two other arms were used to manipulate grasping instruments of varying sizes, and the fourth

arm manipulated the fiberoptic bronchoscope, which was inserted nasally and orally (Figure 2–8). Two simulated intubations were performed with success: the oral intubation was completed in 75 s and the nasal intubation was completed in 75 s.
In another study, a specific system was developed to perform robot-assisted endotracheal intubation, called the Kepler Intubation System (Hemmerling et al., 2012). The system consists of a carbon fiber robotic arm, manipulated through a two-part joystick. At the extremity of the robotic arm, a videolaryngoscope is attached. The system can be controlled through what the authors call an intubation cockpit. The cockpit integrates video feeds laterally to the mannequin's head and from the videolaryngoscope to help manipulate the tip of the laryngoscope.

2.4.1.1 Overview of current airway mannequins

Standard airway mannequins. Many standard airway mannequins are commercially available, and are being used for teaching or research purposes around the world. One of the most widely used airway mannequins is the Laerdal® Airway Management Trainer (Figure 2–9). This model is designed to provide training on the adult airway management. It has fixed dimensions, and comes with left and right lungs and a stomach.

Another model is the Ambu® Airway Management Trainer designed for teaching intubation techniques with all known tubes and supraglottic airway devices (Figure 2–10). The left side of its head is open to allow supervision of the trainee's performance. It integrates acoustic signals that are triggered by excess pressure on the teeth.

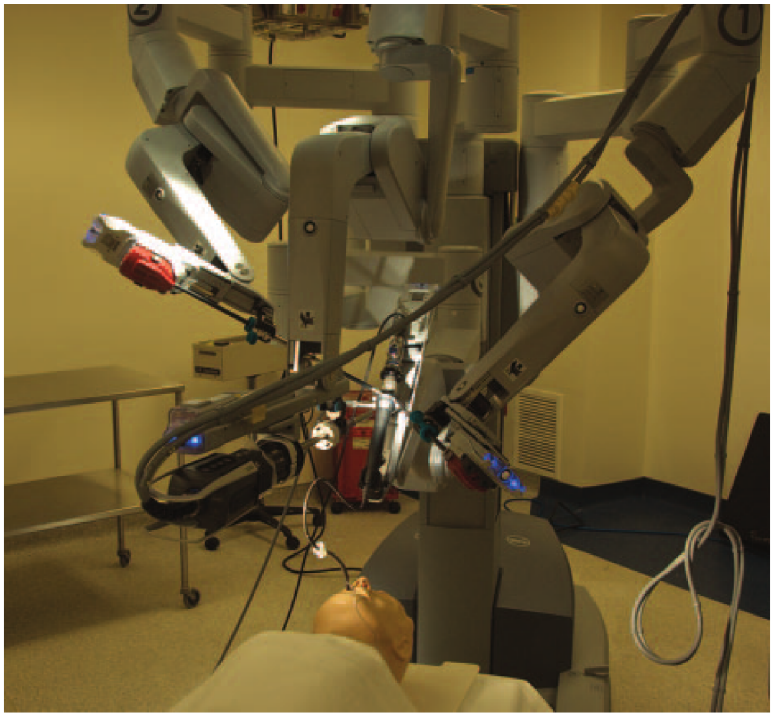


Figure 2–8: Arrangement of the da Vinci Surgical System, bronchoscope, and airway mannequin (Tighe, Badiyan, Luria, Lampotang, & Parekattil, 2010)

Advanced airway mannequins. Advance airway mannequins integrate features in order to provide a more realistic experience for trainees.

The AirSim Advance by TruCorp, shown in Figure 2–11, is one such mannequin. It provides realistic feedback during airway management procedures, it has a 'real feel' skin covering the head, it features an inflatable tongue with real-life size and texture, and has a teeth breakout to simulate the effect of bad practice in laryngoscopy.

The SimMan® by Laerdal is another advanced mannequin. The SimMan is shown in Figure 2–12. This model is a fully computer operated total body simulator, however, for the purpose of this chapter, we will focus on its advanced airway management features. On top of all the features a standard airway management

Figure 2–9: The Laerdal airway management trainer

model has, the SimMan features airway complications such as tongue edema, laryngospasms, posterior pharyngeal swelling, decreased cervical range of motion, and decreased right/left lung compliance.

Adjustable airway mannequins. An adjustable airway model allows toggling between normal and difficult configuration of selected anatomical features to provide a more versatile range of intubation difficulties.

One such system is the parametrically adjustable intubation mannequin developed by Delson and colleagues (Delson et al., 2012) used in the robot-assisted intubation study in the present thesis (Figure 2–13).

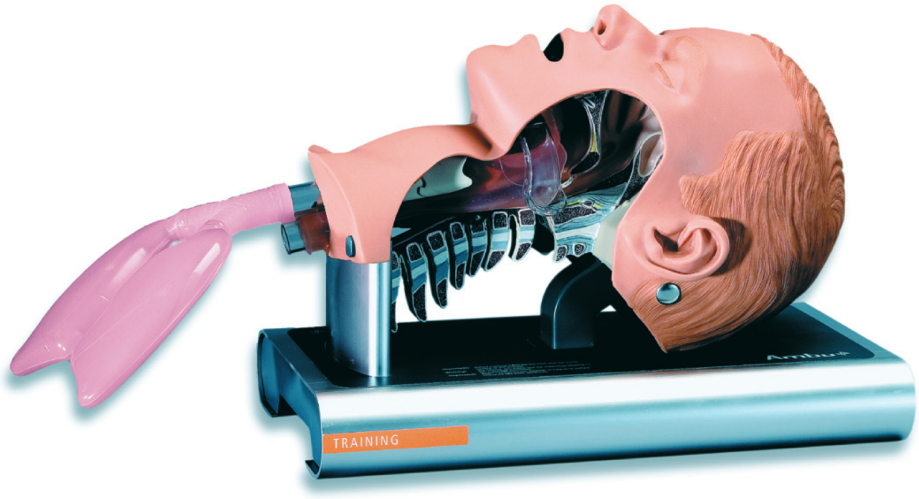


Figure 2–10: Ambu airway management trainer

Figure 2–11: AirSim Advance by TruCorp

This mannequin features separate parts for the skull, face and maxilla, mandible, upper and lower teeth, and the cervical vertebrae from C1 to C7. It allows adjustments in the height of the maxilla (3 levels), the length of the upper incisors (3

Figure 2–12: SimMan by Laerdal

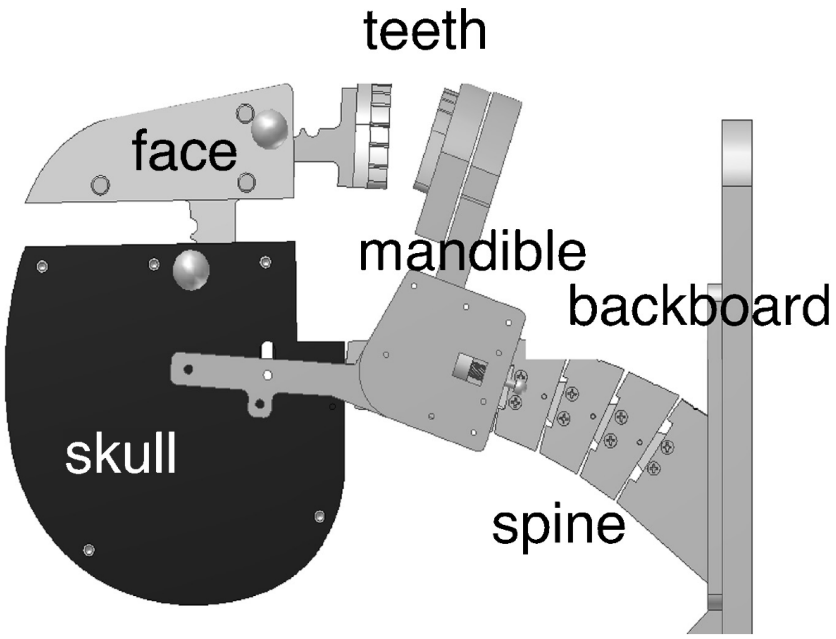


Figure 2–13: Parametrically adjustable airway mannequin (Delson et al., 2012)

levels), the distance of the jaw movement (6 levels), the stiffness of the spine, the

presence or absence of the upper and lower teeth, and the height of the headrest (6 levels).

2.4.2 Manual robots for regional anesthesia

The same group of Tighe *et al.* used the da Vinci system to perform an ultrasound-guided single shot nerve block and a catheter insertion in a regional nerve block mannequin (ultrasound phantom) (Tighe, Badiyan, Luria, Boezaart, & Parekattil, 2010). The ultrasound probe was placed manually, and after the identification of the simulated perineural structures within the phantom, the ultrasound probe was stabilized with the da Vinci system. Using small graspers, the block needle was then picked up from the operating room bed and advanced at a 45-degree angle to the phantom in-line with the ultrasound probe. The setup is shown in Figure 2–14. Although the simulated nerve block was successfully completed, the authors noted that with the current state of the da Vinci system, several steps were not robotically feasible, and "several actions easily completed with the human hand were not so easily mimicked by the robotic grasper" (Tighe, Badiyan, Luria, Boezaart, & Parekattil, 2010).

Another system designed specifically to perform robot-assisted nerve blocks is the Magellan system (Hemmerling et al., 2013b; Morse et al., 2013). This system uses the same components as the Kepler intubation system. The Magellan however has a specifically designed nerve block cockpit and adapter to hold the needle. The robotic arm is controlled with a joystick (as with the Kepler) and can operate at three different speeds. The nerve is identified with an ultrasound machine (held

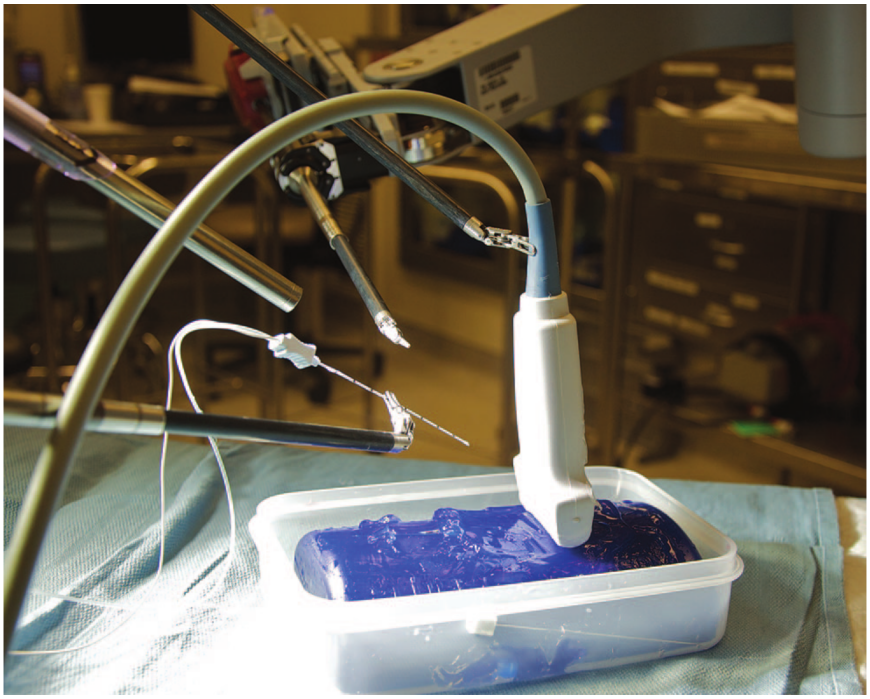


Figure 2–14: Study setup for the robot-assisted regional anesthesia by Tighe, Badiyan, Luria, Boezaart, & Parekattil (2010)

manually), and the needle is advanced using the robotic arm. The group tested their system in 13 patients performing 16 sciatic nerve blocks with 100% success rate.

CHAPTER 3
Technical description

3.1 The pharmacological robots

The pharmacological robot delivers general anesthesia drugs (propofol, remifentanil, and rocuronium) using totally intravenous anesthesia (TIVA). The pharmacological robot is composed of a hardware part and software part. The hardware part is in turn composed of three components, the computer system acting as the brain, the monitors used as input to the system, and the pumps acting as the output (Figure 3–1).

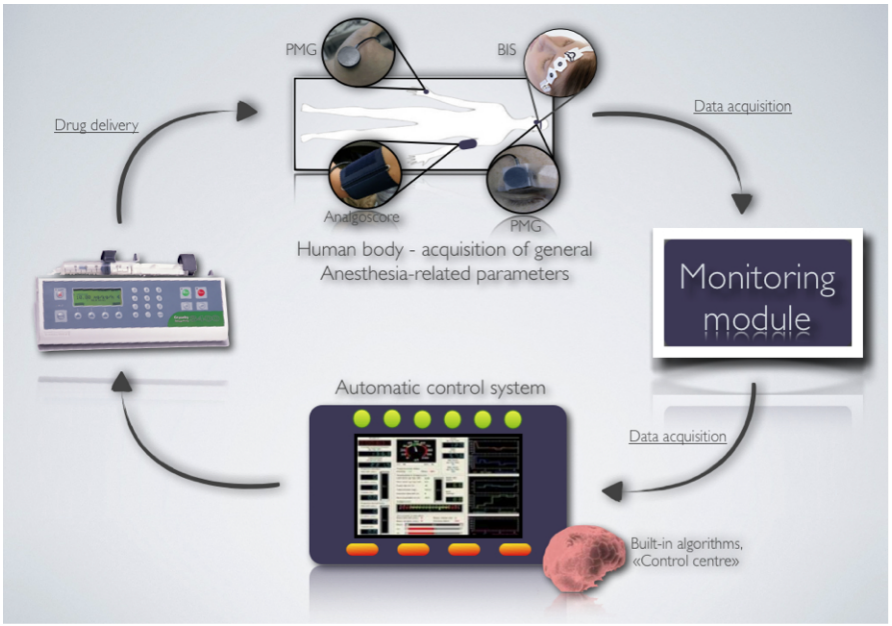


Figure 3–1: Diagram illustrating the closed-loop system.

3.1.1 The hardware components

3.1.1.1 The 'Brain'

The brain of this pharmacological robot is a touchscreen computer (Gateway One ZX, Irvine, CA, USA) running the main software to control the drug delivery. This computer runs the graphical user interface (GUI) of the software. The program running on this computer will be discussed in details in a following section.

3.1.1.2 The monitoring module

The monitoring module shown in Figure 3–1, representing the input of this closed-loop system is composed of vital signs monitors. The parameters monitored are general anesthesia-related parameters: hear rate (HR), mean arterial pressure (MAP), peripheral oxygen saturation (SpO_2), bispectral index (BIS), and muscle relaxation. The Philips Intellivue monitor (Philips, Amsterdam, The Netherlands) was used to monitor the HR, MAP, SpO_2, and BIS.

Bispectral index was used as the control variable for hypnosis to calculate the propofol doses and infusion rates to maintain a preset BIS target value. The heart rate and MAP were used to evaluate the pain level to calculate the effective dose of remifentanil.

In order for the"Brain" to read the monitored parameters, a communication protocol had to be coded in the software in order to allow for a seamless communication with the Philips monitor. This will be discussed in a following section.

The monitoring for muscle relaxation is done using a technique called phonomyography that is based on the fact that a muscle generates a low frequency sound signal when it contracts.

3.1.2 The software part

3.1.2.1 The control software

The control software is developed with LabVIEW 2010 developer suite (version 10.0-32-bit) (National Instruments, Austin, TX, USA).

The control software has at its front end the GUI where the user is able to interact and control the software. The GUI is tabular, and is composed of 4 tabs or pages: the setup page, the induction page, the maintenance page, and the emergence page.

The setup page. The setup page is shown in Figure 3–2. On the setup page, the user enters the patient demographic data, selects surgery type, checks default drug concentrations, and select the drugs for the anesthesia induction.

On this page, the user can also see the status of the communication between the "Brain" and the vital signs monitor, and when the communication is established, the user can start the anesthesia from the setup page.

The induction page. On the induction page (Figure 3–3), the BIS value is shown, as well as the progress of the induction of the drugs (propofol, remifentanil, and rocuronium), in terms of time left and volume remaining.

The induction algorithm flowchart is shown in Figure 3–4. The induction starts with an infusion of remifentanil. When 120 s are elapsed, another dose of remifentanil continues. At the same time, propofol injection starts, provided that propofol induction was selected in the setup page.

At the end of propofol injection, the system waits for 45 s, after which, another dose of propofol is injected if BIS > 60.

An "Inject" button for propofol is integrated to the induction page, to inject a bolus

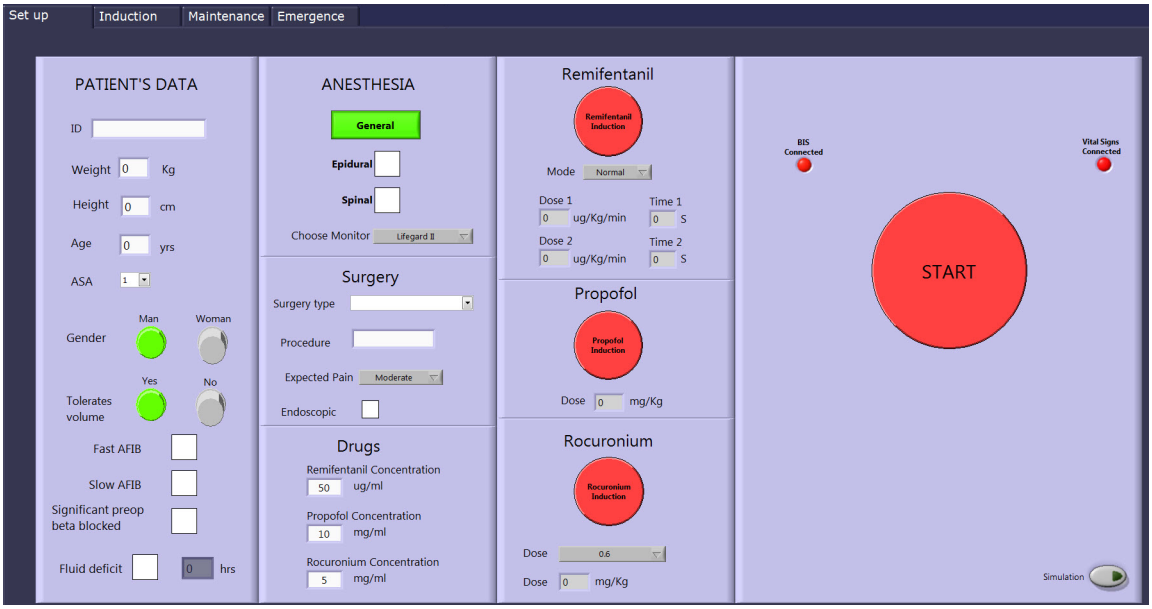


Figure 3–2: The setup page.

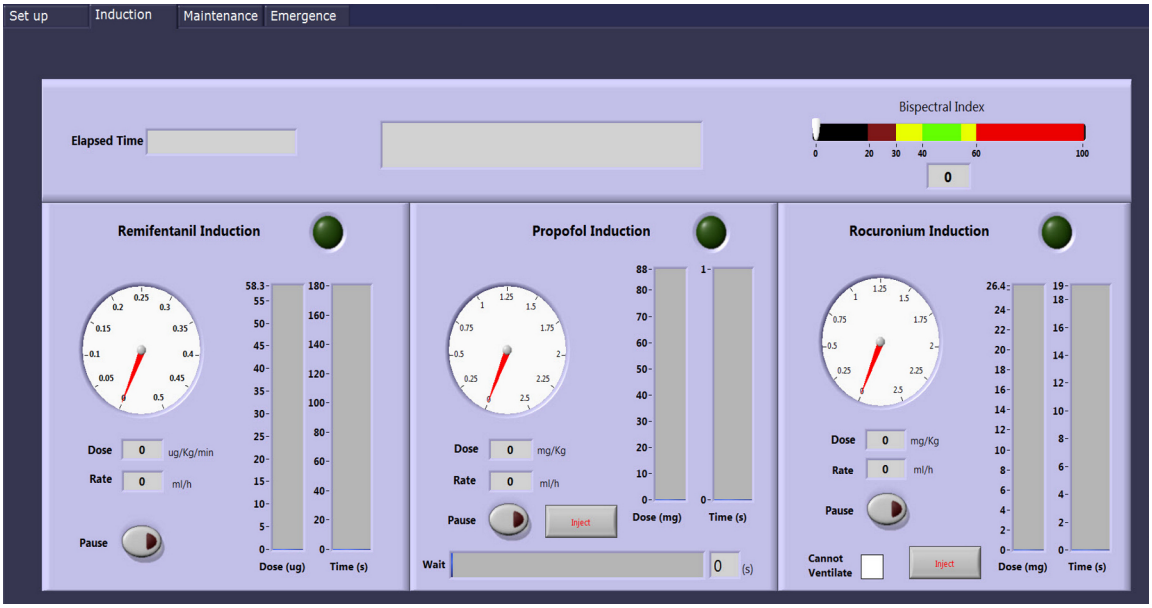


Figure 3–3: The induction page.

of propofol, if BIS is still greater than 70 after the second bolus.

Following propofol induction, the system waits for BIS < 70 (and BIS$\neq$0) to start rocuronium/cisatracurium induction automatically or waits for the user to click the "Inject" button. Rocuronium/cisatracurium induction can be halted by checking "Cannot Ventilate".

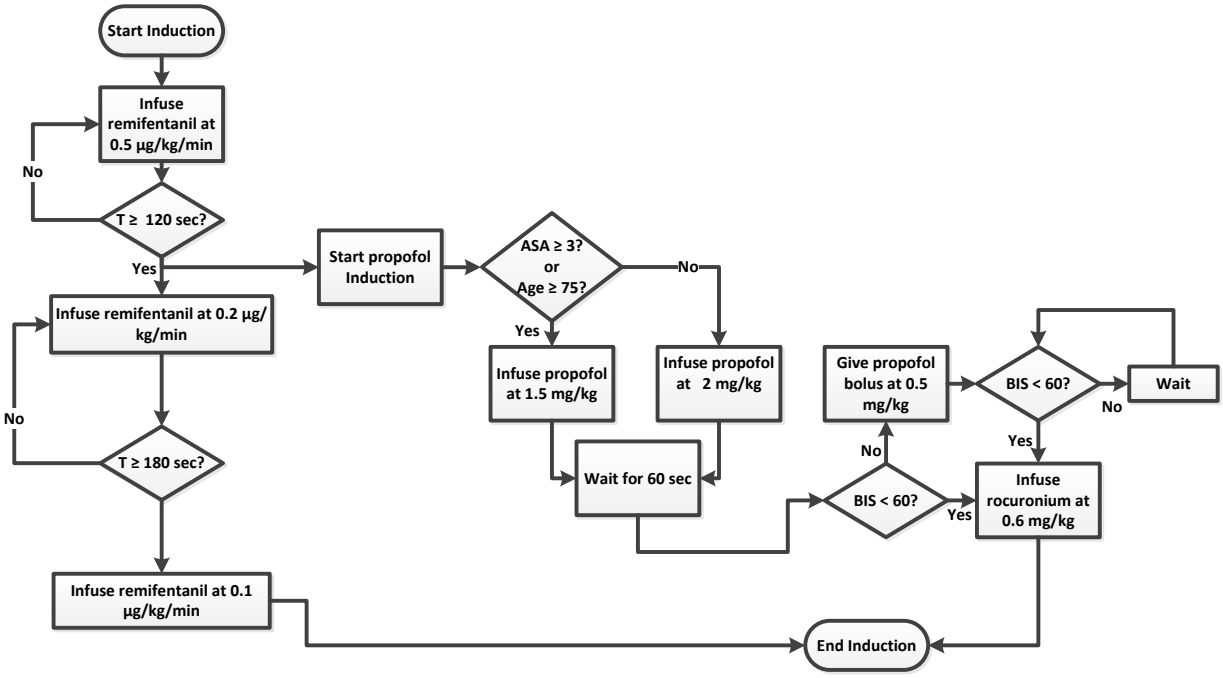


Figure 3–4: The induction phase flowchart.

The maintenance page. The Maintenance page, shown in Figure 3–5, appears automatically after the Induction. The maintenance page displays the BIS values, AS values, continuous infusion rates, drug doses, live video feed, vital signs

(i.e., systolic and diastolic blood pressure, MAP, HR and peripheral oxygen saturation), buttons to administer propofol boluses and rocuronium/cisatracurium boluses, and buttons to indicate definite time points of surgery.

The vital signs (Systolic pressure, diastolic pressure, MAP, HR, and SpO_2) and NociMAP are updated every second.

For safety reasons, the software sets minimal and maximal doses for the infused drugs.

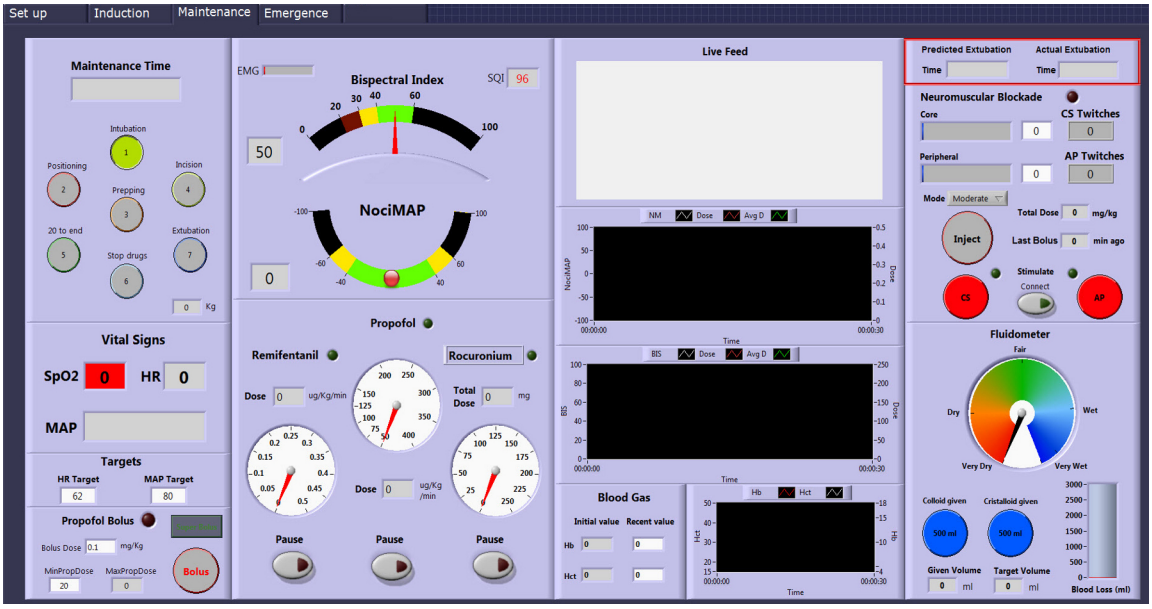


Figure 3–5: The maintenance page.

The minimal and maximal doses of remifentanil are respectively 0.05 and 0.5 μg/kg/min. Remifentanil dose is calculated every 1 minute. The remifentanil dose is 0 when the NociMAP is less than or equal to -44.

The analgesia control flowchart is illustrated in Figure 3–6. The BIS value is updated every 5 s, and the target BIS value is set by default to 45. The minimal propofol dose

is determined from patient data and can be overwritten at all time. The maximal propofol dose is at least 200 g/kg/min and increases automatically if needed. The propofol dose is calculated every 30 s. This dose is 0 when BIS < 20, and it is minimal when $20 <$ BIS < 30. When the BIS is less than 5% of the target BIS, for more than 15 minutes, the minimal propofol dose decreases by 5 μg/kg/min, until it reaches the minimum dose, which is 40 μg/kg/min. A propofol bolus is injected automatically when BIS > 60. There is a delay of 20 s that separates two automatic boluses.

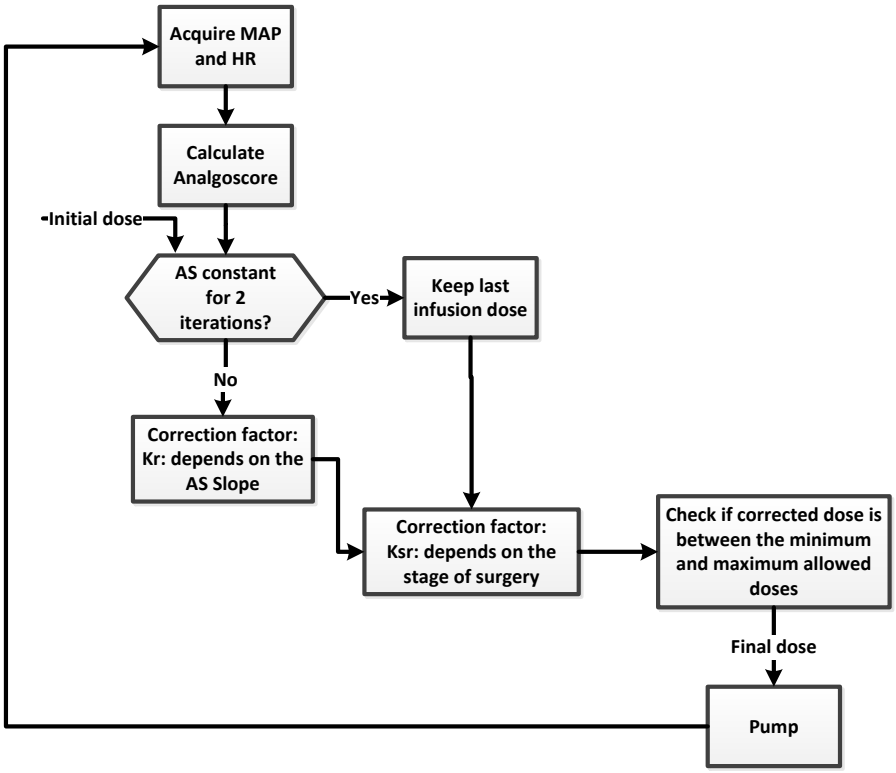


Figure 3–6: The analgesia control flowchart.

A bolus can also be injected manually when the "Bolus" button is pushed. There is no time delay between boluses for the manual mode.

In case the BIS signal is lost (sensor disconnected) or contaminated by artifacts (low SQI, high EMG), the system administers the average propofol dose of the last 5 minutes.

The hypnosis control flowchart is shown in Figure 3–7.

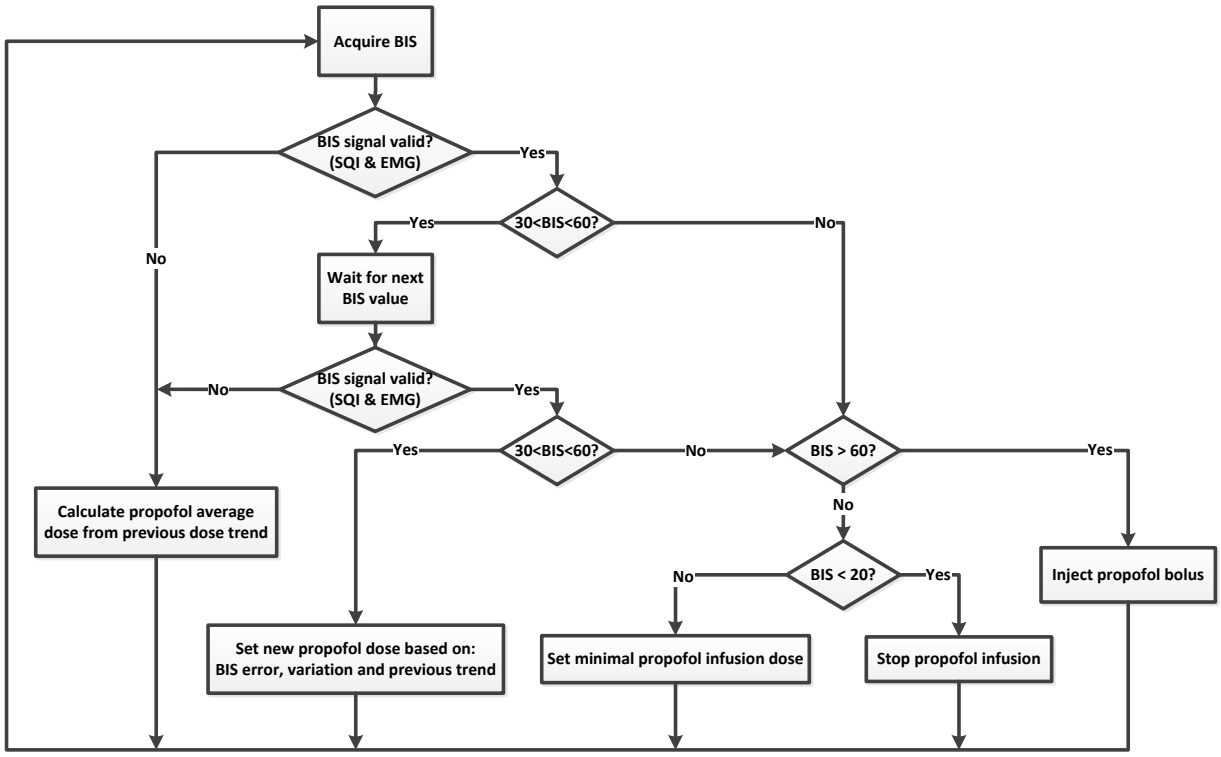


Figure 3–7: The hypnosis control flowchart.

Three modes for muscle relaxation during the anesthesia maintenance are implemented:

- Profound relaxation provides a rocuronium dose of 0.2 mg/kg whenever a bolus is administered
- Moderate relaxation provides a dose of 0.15 mg/kg whenever a bolus is administered
- Superficial relaxation provides a dose of 0.1 mg/kg whenever a bolus is administered.

The neuromuscular blockade control flowchart is shown in Figure 3–8.

On the surgery time point button panel (Figure 3–5), pressing button "Prepping" increases remifentanil infusion before the "Incision" period. Pushing the "15 to end" button disables the emergency bolus and increases the target BIS to 50. The "Stop drugs" button stops all three pumps, while keeping acquisition for BIS and Vital signs for monitoring purposes.

When "Stop drugs" is clicked, the program goes into the emergence page.

The emergence page. The emergence page (Figure 3–9) contains the total doses for propofol, remifentanil and rocuronium or cisatracurium (depending on the drug used). It contains a live feed of the operating field, HR, MAP, SpO_2 and BIS. The emergence page also contains three models for the predicted extubation time.

At the emergence stage, the system is still monitoring the vital signs of patient, and when the BIS is above 60 for more than 30 consecutive seconds, the system instructs the anesthesiologist to extubate the patient.

Finally, there is an "Extubation" button that, when clicked, stops the acquisition from the monitors stops and shuts down the program.

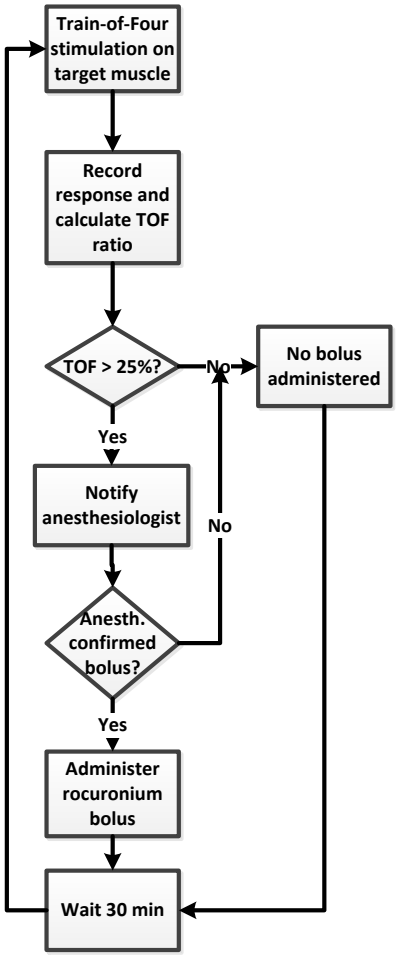


Figure 3–8: The neuromuscular blockade control flowchart.

3.1.3 Communication protocol

3.1.3.1 Philips IntelliVue Patient Monitor

The Philips IntelliVue MX800 (Figure 3–10) was used to monitor and record the patients' vital signs and BIS and acted as an input to the closed-loop system. The Philips IntelliVue Patient Monitor features an integrated PC for one view with clear

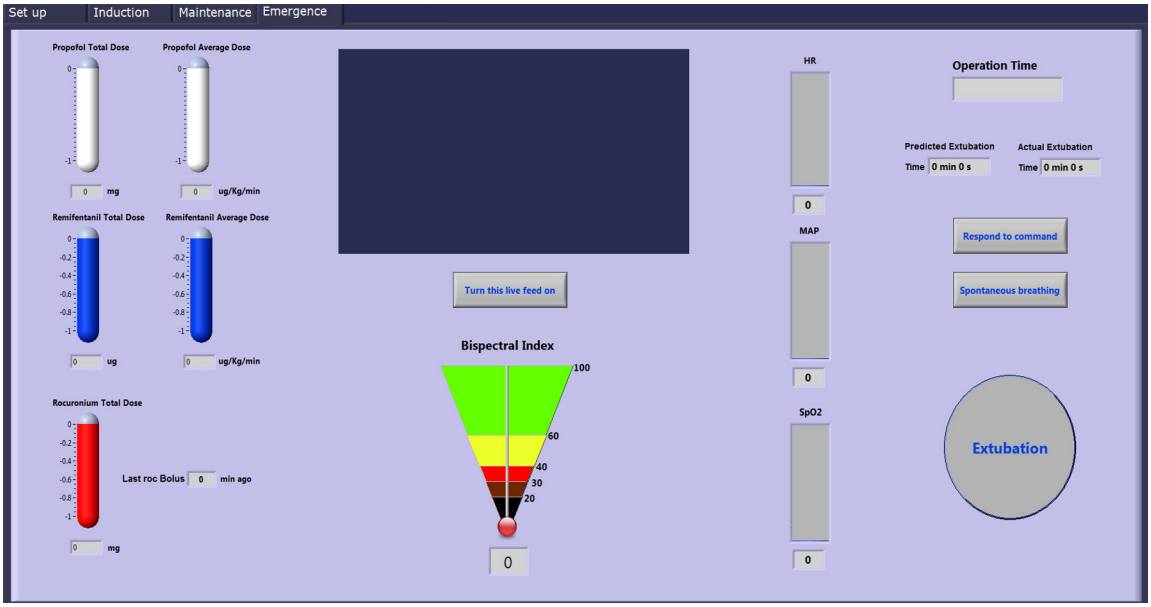


Figure 3–9: The emergence page.

patient status and relevant clinical information for general anesthesia purposes. In order to communicate with the IntelliVue monitor, I used the Data Export Interface provided by Philips. Using this interface, data from the monitor could be transferred via Local Area Network (LAN) to an external computer. The following data can be transferred using the IntelliVue Data Export:

- All measurement numerics and alarm data (real-time update rates up to 1024 ms)
- Wave data
- Intellivue monitor system data.

The communication protocol was written in C# (Microsoft, Redmond, WA, USA) using the Data Export Interface. The communication protocol code is shown in Appendix A.

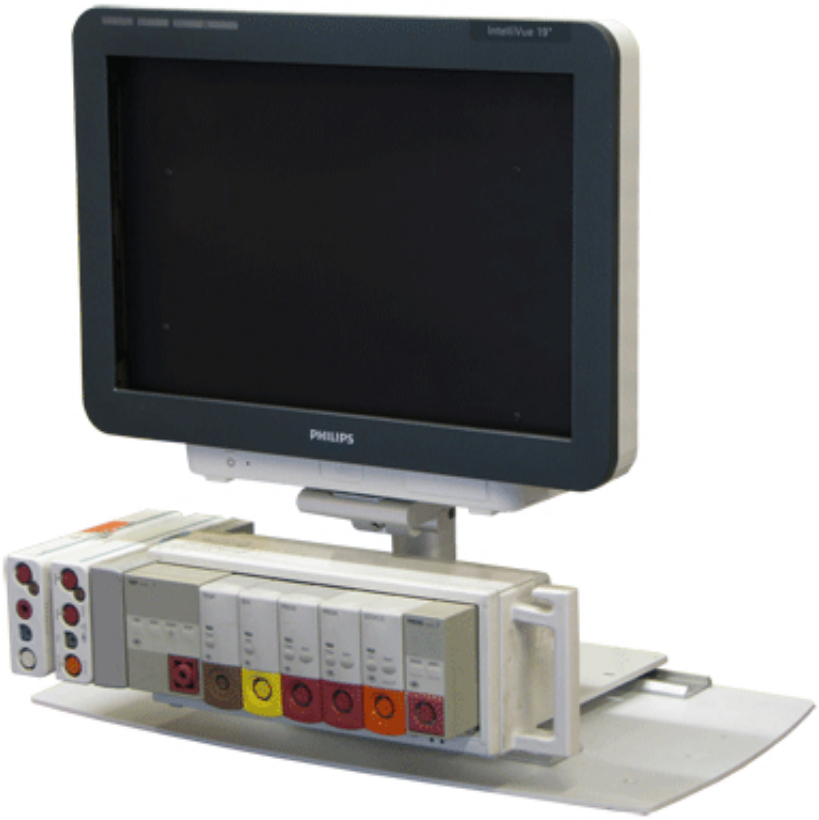


Figure 3–10: Philips IntelliVue MX800 patient monitor.

The protocol dialog between the IntelliVue monitor Data Export server and the computer in illustrated in the diagram in Figure 3–11.

The developed communication protocol library was then called in the LabVIEW program in order to allow communication between the "Brain" and the monitor.

3.1.3.2 The Graseby pump

The Graseby 3400 syringe pump was used to deliver the anesthetic drugs. This pump can use a wide range of syringes (5-60 ml), and can deliver maintenance flow rates in the range of 0.1-400 ml/hr in steps of 0.1 ml/hr. It has a bolus option that

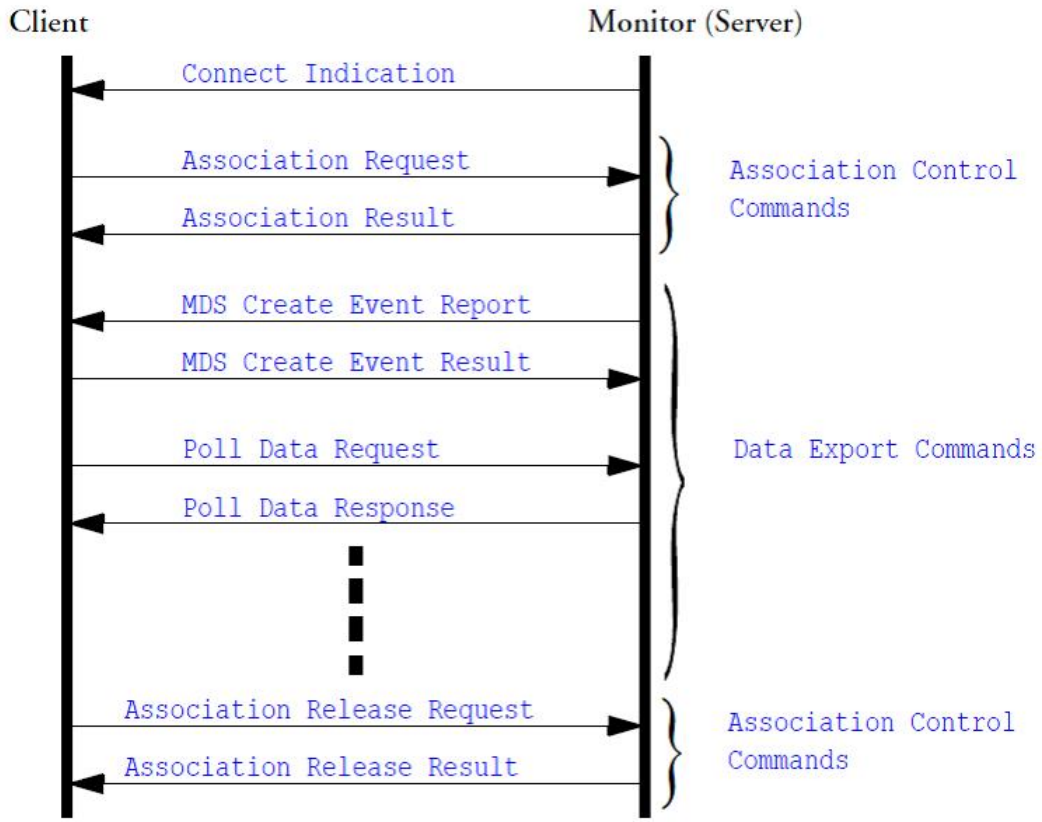


Figure 3–11: Protocol dialog between the IntelliVue Data Export server and the computer client. (Philips, 2011)

allows for a flow rate up to 1200 ml/hr.

The 3400 pump can be driven by a computer via the serial port, using a maximum bit-rate of 9600 bps.

The communication protocol was written in LabVIEW since it has built in serial communication functions.

A serial communication protocol LabVIEW code snippet is shown in Figure 3–12.

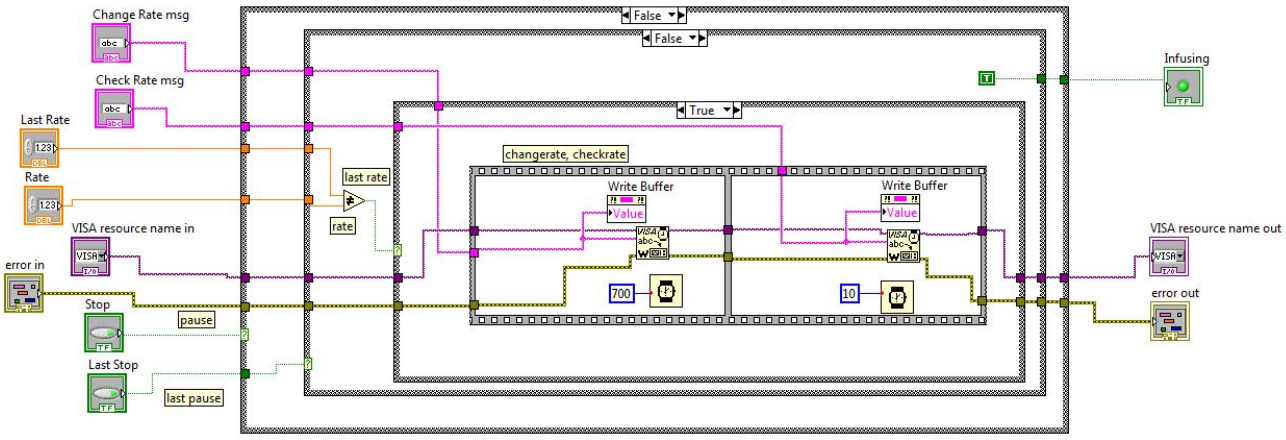


Figure 3–12: Serial communication code snippet from LabVIEW.

3.1.4 Tele-anesthesia

The automated anesthesia drug-delivery system was used in a tele-anesthesia application in order to administer general anesthesia remotely from Montreal to Pisa. The tele-anesthesia application consists of the following components: audio-video communication with the local health care provider, video feed of the important monitoring systems including the surgical site, and remote control of the locally installed anesthesia system.

The remote set-up and local set-up were composed of a master-computer (Montreal), an audiovideo-purpose computer (both sites), and a slave-computer (Pisa), respectively (Figure 3–13).

The remote center (Pisa) computer set-up was composed of two computers. The automated anesthesia delivery program was installed on one computer to control delivery of the anesthetic drugs ('slave system') automatically via Graseby 3400 pumps.

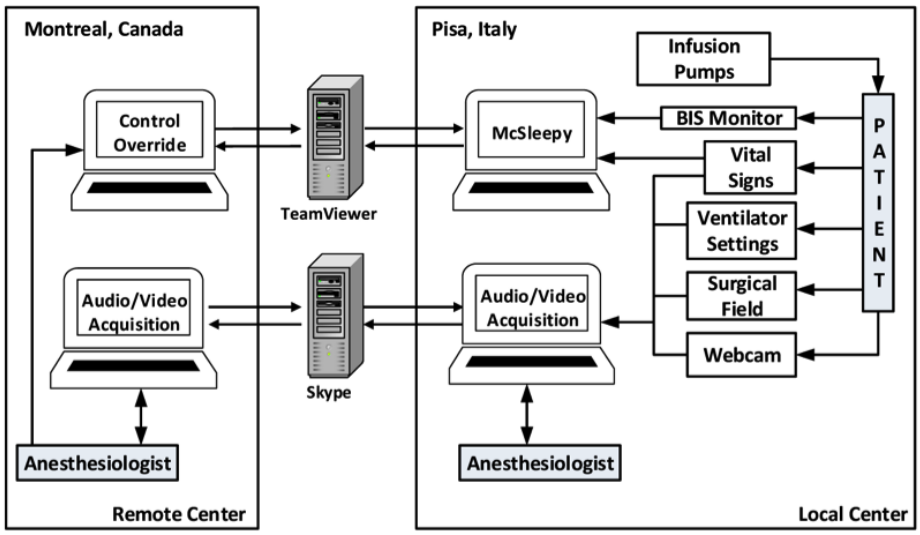


Figure 3–13: Diagram showing the transcontinental anesthesia set-up and operation.

On a second computer, live feeds of four webcams were displayed for different monitoring purposes: automated anesthesia delivery system interface, view of the surgical field, imaging of the glide scope during intubation, and vital sign monitor.

The local center consisted of two computers. The automated anesthesia drug delivery system control program was installed on one computer ('master system'); communication and control of the 'slave system' was done using the software TeamViewer. The second computer was connected with the video-feed remote computer in Pisa via Skype. Both local computer systems and remote computer systems were connected via a standard Internet connection with a high bandwidth of up to 8 Mbps.

3.1.5 The user interface

As mentioned above, the control system's user interface is developed in LabVIEW and it integrates 4 pages: setup, induction, maintenance, and emergence. The user interface was designed according to intuitive anesthetic interface guidelines (Charabati et al., 2009). The user interface incorporates an integrated anesthesia monitor that presents information about the three components of anesthesia, helping in assessing general anesthesia more clearly, and helping anesthesiologists to detect adverse events faster. The user interface also includes trends for vital signs and anesthetic drugs dosages.

3.2 The Kepler Intubation System

Similar to the pharmacological robot, the Kepler Intubation System (KIS) also has a hardware component and a software component.

3.2.1 The hardware components

The KIS hardware part is composed of robotic arm (JACO, Kinova, Montreal, QC, Canada), a joystic (ThrustMaster T.Flight Hotas X, Guillemot Inc., New York, NY, USA), the software control system, and a videolaryngoscope (Pentax AWS, Ambu A/S, Ballerup, Denmark).

The JACO arm allows movement around 6 degrees of freedom with unlimited rotation on each axis. Each of the 6 degrees of freedom of the arm is mapped to a button or axis of the joystick.

The structure of the JACO arm is made of carbon fiber, and is mounted on a custom made cart that holds it, the software control system, the joystick, and a power backup system.

The videolaryngoscope used is the Pentax-AWS airway scope. It is an indirect videolaryngoscope that integrates tube guidance. The intubation process is visualized on a built-in LCD through a camera that reaches the tip of laryngoscope. The camera is attached to the laryngoscope handle through fiber optics.

Figure 3–14: JACO arm, Pentax-AWS, and the joystick.

Figure 3–14 shows the Pentax videolaryngoscope connected to the JACO arm along with the joystick used to control the arm.

3.2.2 The software components

The JACO arm has an application programming interface (API) for C# that allows access for a complete library of functions. The API is linked to the arm through USB 2.0 connector.

The control software is written in C#, and allows control of all the axes and all 6 degrees of freedom through the joystick. The system can also acquire details about the location and status of the through the provided API.

The user interface of the control system for the KIS is developed in LabVIEW and is shown in Figure 3–15.

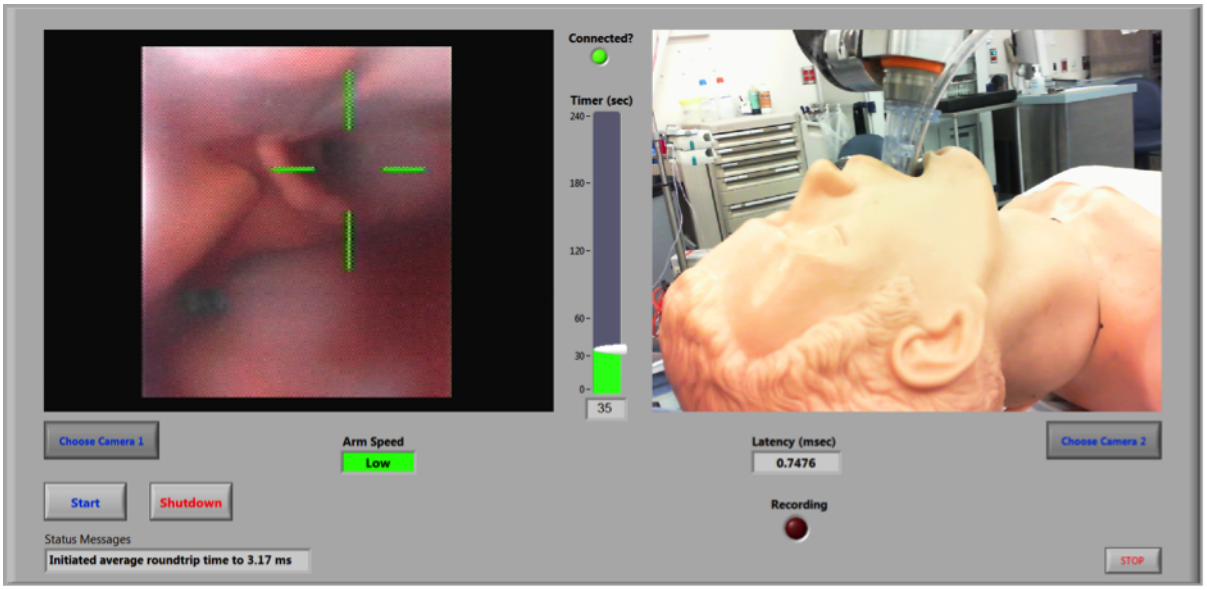


Figure 3–15: Cockpit of the Kepler Intubation System.

The KIS user interface contains 2 video feeds, one coming from the videolaryngoscope (Figure 3–15 top left) showing its position in the mouth, the other from a webcam placed laterally to the airway mannequin (Figure 3–15 top right) showing

the position of the JACO arm and the videolaryngoscope in relation to the airway mannequin. Between these two videos is a visual indicator of the time elapsed since the start of the intubation: this control turns yellow or red after 2 or 3 minutes have elapsed, respectively, as intubation is a time sensitive procedure. The network latency is also displayed and color-coded: green for latencies less than 200 ms, yellow for latencies between 201 and 400 ms, and red for latencies greater than 400 ms. Additionally, the arm speed is color-coded: green for low speed (used while the laryngoscope is inside the patient's mouth), yellow for medium (used to descend the laryngoscope towards the patient's mouth), and red for high speed (used to move the laryngoscope initially into position).

3.2.2.1 The control scheme

The control scheme is depicted in Figure 3–16. Each of the six degrees of freedom of the robotic arm are mapped to a button or axis of the joystick. The six degrees of freedom allow for the system to mimic all of the possible movements of the human wrist: flexion, extension, radial and ulnar deviation, supination, and pronation. The trigger of the joystick is used to mark the beginning of the intubation procedure. Also, a button is mapped to begin and stop recording movements of the system. Additionally, a button is mapped to begin and stop recording movements of the system. Two buttons on the top of the joystick are used to either increase or decrease the speed of the robotic arm. The left-right and forward-reverse axes of the primary joystick handle are mapped to their corresponding movements of the robotic arm. Twisting the primary joystick handle in either direction will rotate the video laryngoscope in a similar fashion. The throttle control of the joystick is used

to change the vertical position of the video laryngoscope. Finally, the hat switch controls the rotation of the tip of the video laryngoscope.

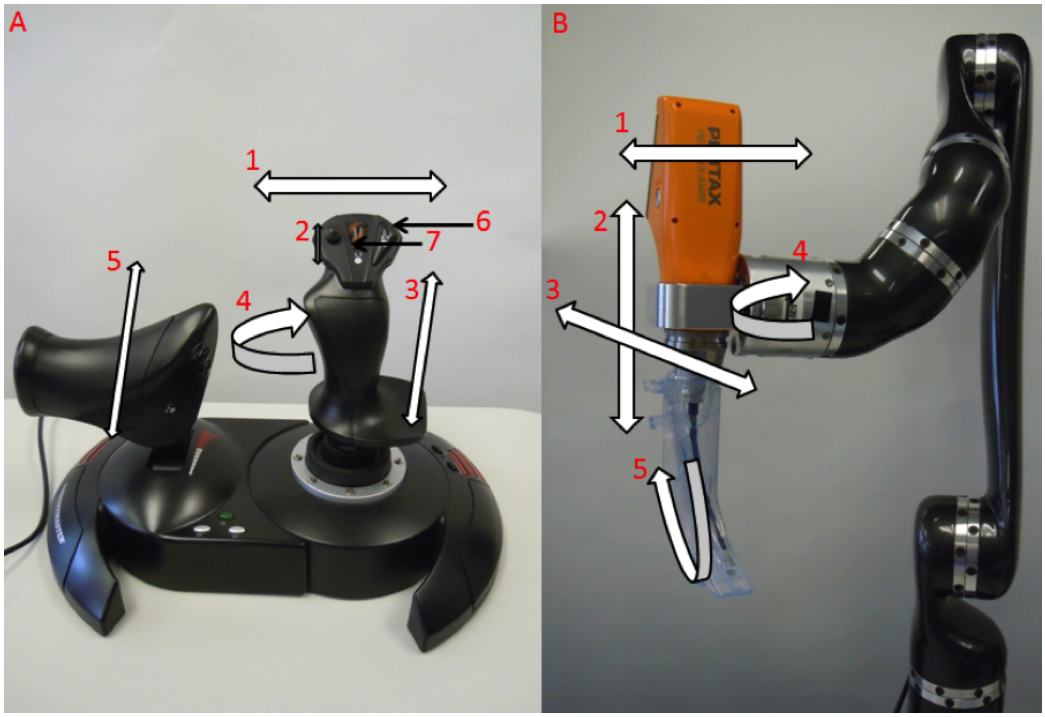


Figure 3–16: Control scheme of the KIS. (A) Movements of the video laryngoscope; (B) corresponding movements of the joystick. 1, left and right; 2, up and down; 3, forward and backward; 4, rotation of the blade horizontally; 5, rotation of the tip of the video laryngoscope; 6, decreasing the speed; 7, increasing the speed.

3.2.2.2 The semi-automated mode

Semi-automation is a mode used in the KIS in order to perform robot-assisted intubation in a semi-automated fashion. This is done by recording the movements necessary to perform an intubation, then replaying the recorded movements.

CHAPTER 4
Article 2

McSleepy: A Closed-Loop Automated Anesthesia Drug Delivery System

Mohamad Wehbe*, Erik Arbeid†, Shantale Cyr‡, Pierre A. Mathieu#, and Thomas M. Hemmmerling‡#

*Department of Experimental Surgery, McGill University, Montreal, Canada

†Department of Anesthesiology, University of Pisa, Pisa, Italy

‡Department of Anesthesia, McGill University, Montreal, QC, Canada

#Institute of Biomedical Engineering, University of Montreal, Montreal, Canada

Keywords: McSleepy, automated anesthesia, closed-loop system, general anesthesia, and teleanesthesia.

This Article has been published in the Journal of Clinical Monitoring and Computing, February 2014, Volume 28, Issue 1, pp 27-34.

Contribution of Authors

- Mohamad Wehbe: first draft of manuscript, data analysis, preparation of figures and tables
- Erik Arbeid: data analysis, and system setup.
- Shantale Cyr: revision of manuscript and artworks.
- Pierre Mathieu: revision of manuscript.
- Thomas Hemmerling: technology development, supervision of contents, revision of manuscript versions.

Abstract

To control the three components of general anesthesia (hypnosis, analgesia, and neuromuscular blockade) an automated closed-loop anesthesia drug delivery system (McSleepy) was developed. Bispectral index was used as the control variable for hypnosis, the analgoscore for analgesia and phonomyography for neuromuscular blockade, respectively. McSleepy can control the induction, maintenance and emergence from general anesthesia. To do so, a large touch screen is used to provide a user friendly interface, permitting bidirectional communication: the user giving information about the different stages of anesthesia, and the system prompting the anesthesiologist to perform certain actions such as mask ventilation, intubation or waking-up the patient using audio clips with voice commands. Several safety features are implemented to provide a secure and reliable anesthesia. The system was tested on 15 patients undergoing elective surgery requiring general anesthesia. Evaluation of McSleepy was done through an assessment of its clinical performance and using Varvel's performance indices. The system was found to be clinically useful by providing good precision in drug administration and reliable for the duration of a general anesthesia.

4.1 Introduction

General anesthesia is achieved through the use of drugs to control hypnosis (the loss of consciousness), analgesia (the reduction of pain sensation), and neuromuscular blockade (Miller, 2009). Administration of those drugs is generally under the manual control of an anesthesiologist.

Over the last two decades, closed-loop systems have been developed and used in research focused on comparing advantages and disadvantages of closed-loop systems

versus manually administered anesthetic drugs. Closed-loop systems consist of a computer, the "brain" of the system, which is used to run software in order to determine the right dose of a drug at the right time depending on the value of a controlling parameter used to monitor a target value. Expert systems are based on pharmacodynamic and pharmacokinetic considerations, the output of which acts, in general, on an infusion pump which is used to deliver the drug. Depending on the input parameter and the amount of injected drug, the offset of the actual value of the monitored parameter from the target value is evaluated at meaningful time intervals and administration of the drug is adjusted accordingly without human intervention. So far, closed-loop systems have been mainly used for the control of hypnosis (Morley et al., 2000; Struys et al., 2001) and muscle relaxation (Eleveld et al., 2005) due to the absence of a meaningful monitoring parameter of pain during unconsciousness. Some researchers have used hemodynamic parameters, such as blood pressure, as an indirect measurement of pain or nociception (Gentilini et al., 2002). The Analgoscore (AS) uses both heart rate (HR) and blood pressure to represent the level of nociception (Hemmerling et al., 2009). This score was integrated in our closed-loop system for the monitoring of pain.

We present McSleepy as the first completely automated expert-based anesthesia delivery system, which automatically controls all three components of general anesthesia, as well as all its phases, i.e. induction, maintenance and emergence from anesthesia.

4.2 Materials and Methods

4.2.1 System Specifications

The bispectral index (BIS), measured using the BIS Vista^{TM} monitor (Aspect Medical Systems, MA, USA), was used as control variable for hypnosis to calculate the propofol doses and infusion rates to maintain a preset BIS target value. The Analgoscore was the control variable used for evaluation of the pain level. It was derived from HR and mean arterial pressure (MAP), measured using a vital signs monitor (CASMED 740, CAS Medical Systems Inc. Branford, CT, USA), and its value affected the effective dose of remifentanil. To determine the depth of the neuromuscular blockade, phonomyographic signals from the adductor pollicis muscle (AP) and from the corrugator supercilii muscle (CS) were used by McSleepy to compute the train-of-four (TOF) ratio and administer the required dose of rocuronium. The three infusion pumps (Graseby 3400, Graseby Medical, UK) used as actuators for the injection of the three drugs were under the control of the computer via RS-232 ports.

During induction, the remifentanil infusion dose was 0.5 μg·kg^{-1}·min^{-1} for the first 2 minutes, 0.2 μg·kg^{-1}·min^{-1} for the third minute, and 0.1 μg·kg^{-1}·min^{-1} for the remainder of the induction.

As for propofol, its induction started 2 minutes after the beginning of the remifentanil induction with a dose varying from 1.5 to 2 mg·kg^{-1}, depending on patients' characteristics (patient's ASA and age). After the propofol bolus, the system waited for the BIS to drop below 60 for a period of 60 s, after which rocuronium bolus of 0.6 mg·kg^{-1} was then given. The induction is illustrated in Fig. 4–1.

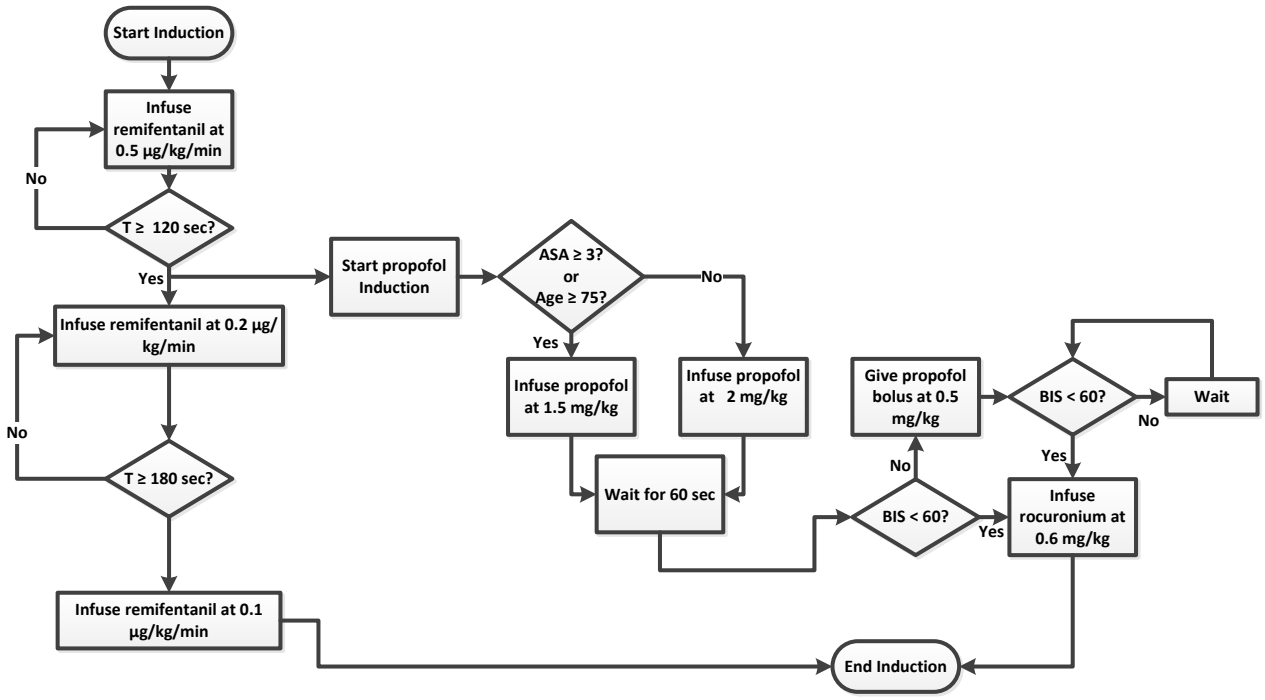


Figure 4–1: Flowchart illustrating the induction phase.

This is an expert based system that emulates the decision-making ability of an experienced anesthesiologist.

4.2.1.1 Hypnosis Control

The BIS served as the control variable for hypnosis control. Relying on this single input signal, the controller could be misled by artifacts that might occur within the electroencephalographic (EEG) signal, posing safety risks to the patient (Locher et al., 2004). To assess the reliability of a BIS value, a signal quality index (SQI) and the electromyograph (EMG) are provided on commercially available BIS monitors. The signal quality index reflects the percentage of time the EEG signal was

artifact-free over the last minute (Kelly, 2003). Artifacts contaminating the EEG are mainly high-frequency signals induced by the use of surgical instruments or due to EMG activity.

The hypnosis control was based on the flowchart illustrated in Fig. 4–2. The system acquired the BIS, SQI and EMG every 5 s and calculated a moving average of the valid BIS every 20 s. A BIS measurement was assumed valid when the SQI $> 40\%$ and the EMG is < 40 dB. If the BIS average was between 30 and 60, the system would set the propofol dose based on the BIS error (difference between the current BIS value and the target, i.e. 45). If the BIS average was between 20 and 30, the system would administer the minimal propofol dose, and if the BIS average was greater than 60, an automatic propofol bolus was given. Furthermore, if the BIS average was less than 20, propofol infusion was stopped.

The propofol dose calculation is a function of the previous dose and the adjustment factors are proportional to the BIS error (the difference between the actual BIS value and the target), the BIS variation (the difference between two consecutive BIS values), and the BIS trend (the difference between the target and the BIS average over the last 5 minutes).

4.2.1.2 Analgesia Control

The Analgoscore, a pain score derived from HR and MAP, was used as the control variable to titrate the effective dose of remifentanil. Its scale ranges from -9 (very profound analgesia) to +9 (very superficial analgesia) in increments of 1, with a target range of -3 and +3 for optimal analgesia (Hemmerling et al., 2009). The

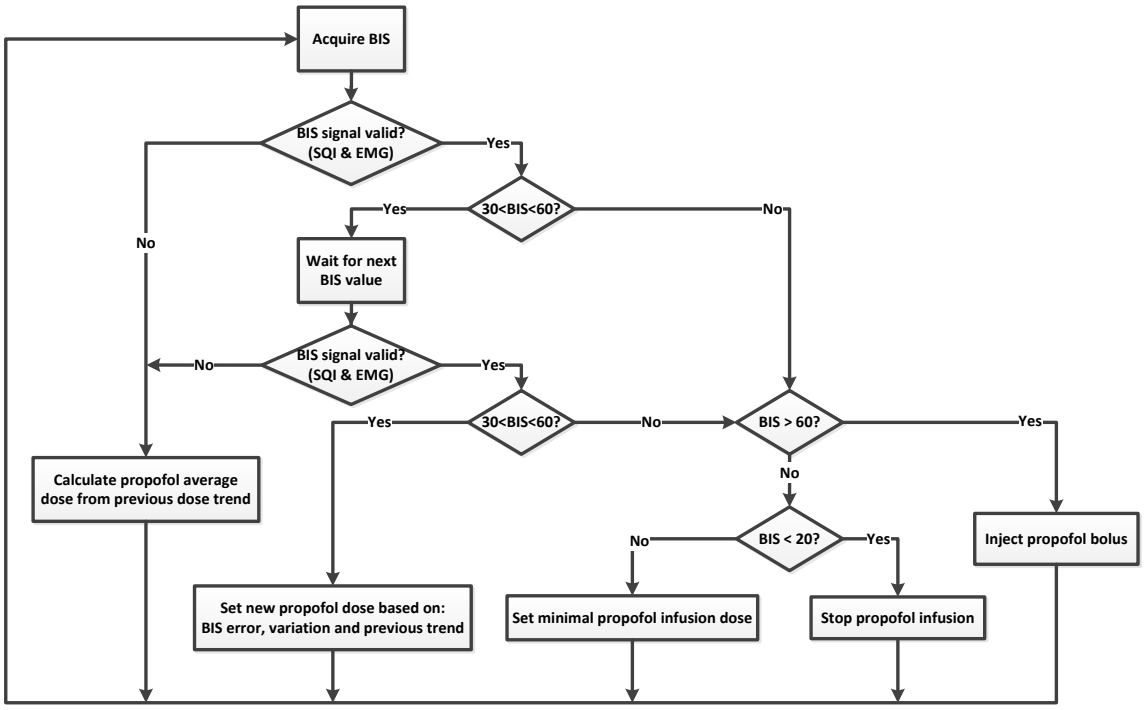


Figure 4–2: Flowchart describing the hypnosis control: bispectral index (BIS) values are acquired every five seconds, and they are averaged every 20 sec. The BIS error represents the difference between the BIS value and the target value. The BIS variation is the difference between the present and previous values.

remifentanil dose was calculated from the following equation:

$$Dose_{remi} = previousDose_{remi} \times K_r \times K_{sr} \tag{4.1}$$

where K_r is a coefficient proportional to the difference between the last and present AS values, while K_{sr} is a coefficient that is dependent on the stage of surgery.

The analgesia is illustrated in the flowchart in Fig. 4–3.

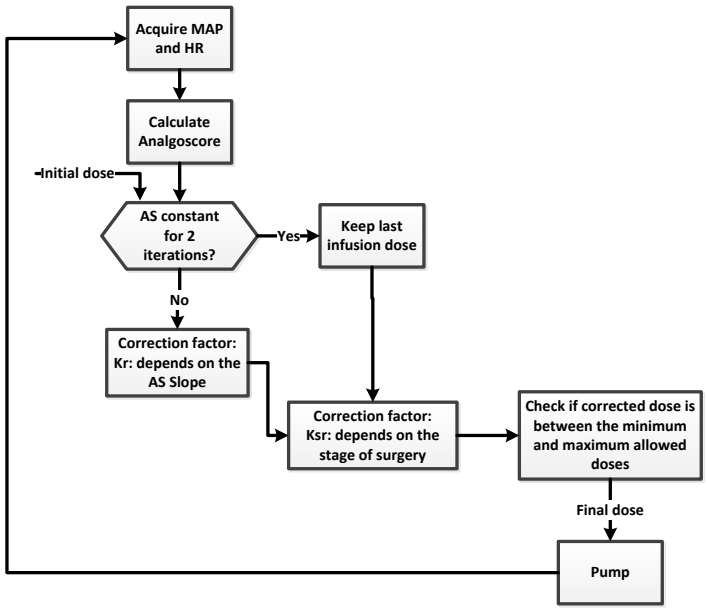


Figure 4–3: Flowchart describing the analgesia control: the Analgoscore is determined based on the HR and MAP. Correction coefficients are calculated based on the variation of the score over time and the stage of surgery.

Since MAP and HR can be influenced by factors other than analgesia, hypovolemia was defined as an increase in HR with no increase in MAP, and vagal reactions were defined as a decrease in HR with no decrease in MAP. When such reactions occurred, the system administered a preset minimal dose of remifentanil.

4.2.1.3 Neuromuscular Blockade Control

Phonomyography was used to monitor the depth of neuromuscular blockade in patients during surgery. Train-of-four stimulation is one of the most widely used patterns for monitoring neuromuscular functions (Viby-Mogensen, 1982). It consists of delivery of four square-wave stimuli of 0.2 ms duration at a frequency of 0.5

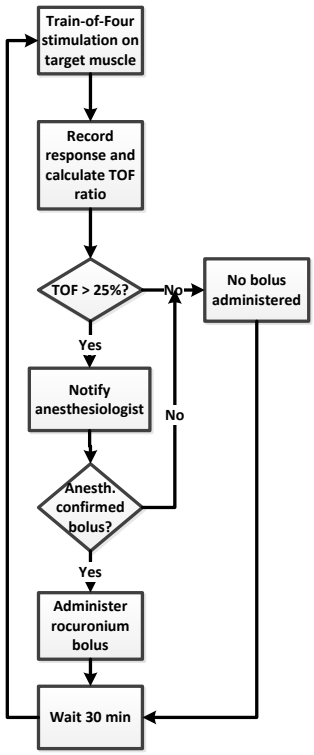


Figure 4–4: Flowchart describing the neuromuscular blockade control: a rocuronium bolus is administered automatically based on the TOF ratio. A TOF stimulation is done every 30 min.

Hz with a peripheral nerve stimulator. The anesthesiologist defined a neuromuscular blockade mode based on the patient's pre-operative data: "Moderate" or "Profound", and the system administered a bolus of 0.15 or 0.2 mg·kg^{-1} of rocuronium for every TOF-ratio greater than 25%, depending on the predefined mode. The neuromuscular blockade control is illustrated in Fig. 4–4.

4.2.2 User Interface

The McSleepy graphical user interface was developed using LabVIEW (National Instruments, TX, USA). The user interface is composed of 4 windows: i) a set-up window where the patients' characteristics (sex, age, weight, ASA, type of surgery) are entered along with drug concentrations; ii) an induction window that shows the BIS values and the progress of the induction of the drugs (propofol, remifentanil, and rocuronium) and contains control buttons to pause the drug infusion at any given time; iii) a maintenance window that displays the BIS values, AS values, continuous infusion rates, drug doses, live video feed, vital signs (systolic and diastolic blood pressure, MAP, HR and peripheral oxygen saturation), buttons to administer propofol boli and rocuronium boli, and buttons to indicate definite time points of surgery; iv) an emergence window which appears after discontinuation of the anesthetic drugs that shows the extubation time, total drug consumption, BIS values and vital signs. The four different windows are shown in Fig. 4–5.

An important feature in McSleepy is the ability of the anesthesiologist to inform it of the different stages of surgery: intubation, positioning, prepping, incision, and 20 min to end. This is done using push buttons on the maintenance window Fig. 4–5(c). This helps McSleepy to adapt the drug infusion doses according to different time points in the surgery.

Voice clips are also part of the user interface. On the induction window, the system will alert audibly the anesthesiologist about the different stages of induction by saying "remifentanil started", "propofol started" and "rocuronium started" in their respective order. Also during the induction, when the BIS dropped below 85

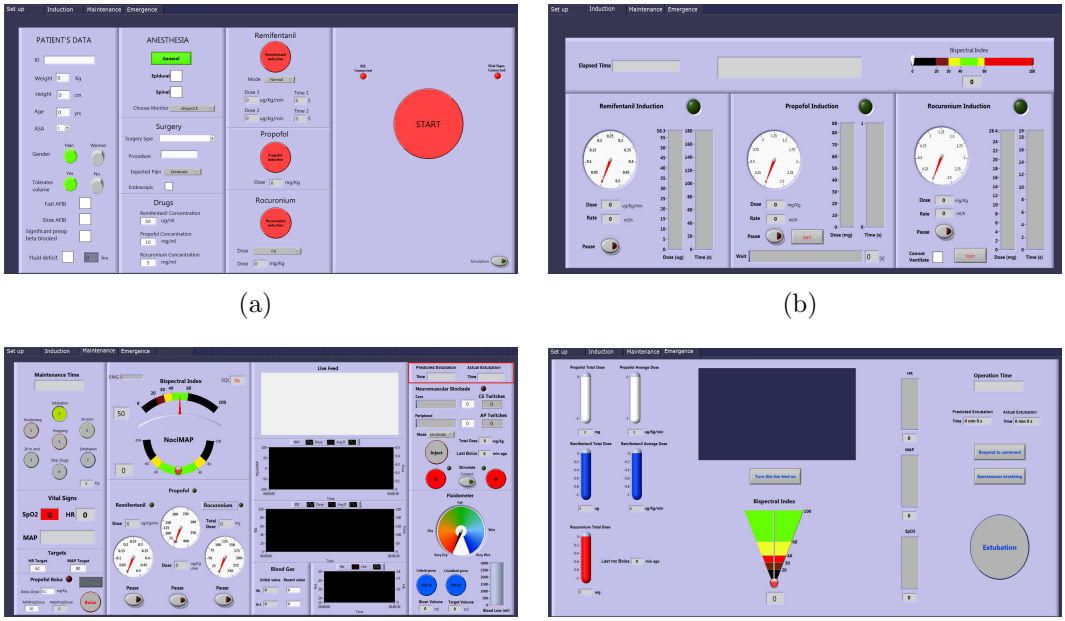


Figure 4–5: Graphical user interface of the McSleepy system: (a) the setup interface: patient data are entered in the first columns, information about anesthesia and surgery type is entered in the second column, and information about the induction is entered in the third column, (b) the induction interface: seperated into three columns containing information about the drugs being induced, (c) the anesthesia maintenance interface: contains information about the different stages of surgery, displays the vital signs and drug doses (d) the emergence interface: displays the total doses of drugs, the BIS value and vital signs.

and before the infusion of rocuronium, the system will ask the anesthesiologist to "please ventilate" the patient. On the maintenance window, and after 75 s of the end of the induction, the system will ask the anesthesiologist to "please intubate" the patient. Finally, after the interruption of drug infusion at the end of surgery, when BIS was over 60 for more than 30 consecutive seconds, the system would tell the anesthesiologist to "wake-up the patient".

4.2.3 Safety Features

In order to provide secure and reliable anesthesia, safety features were implemented in the closed-loop system. In the induction phase, the system continuously checked the patient's BIS and did not administer rocuronium (muscle relaxant) before the patient was totally unconscious (BIS < 60). In addition, acoustic signals (voice commands) were integrated to notify the anesthesiologist of the different periods of induction, asking him or her to ventilate the patient when the BIS dropped below 85, and intubate the patient after 75 s of the end of the induction.

In the maintenance phase, minimum and maximum limits were set for the administered drugs in order to ensure that the closed-loop system did not under- or over-anesthetize the patient.

The closed-loop system would stop propofol and/or remifentanil infusion whenever the BIS and/or the AS (HR and MAP) reached a preset lower threshold, respectively. This was done in order to avoid any cardiovascular side effects.

A rocuronium bolus was given to patients whenever the TOF-ratio was greater than 25%, however the system would wait for a period of at least 5 minutes between two consecutive rocuronium boli. Also a lockout period of 20 minutes before the end of surgery was chosen, during which the system did not give any additional rocuronium to avoid a too profound neuromuscular blockade delaying extubation, but could be manually overridden.

In case of a cable disconnection, for example if the connection was lost between an infusion pump and the anesthesia-control computer, an alarm would sound to alert the local user to reconnect the pump.

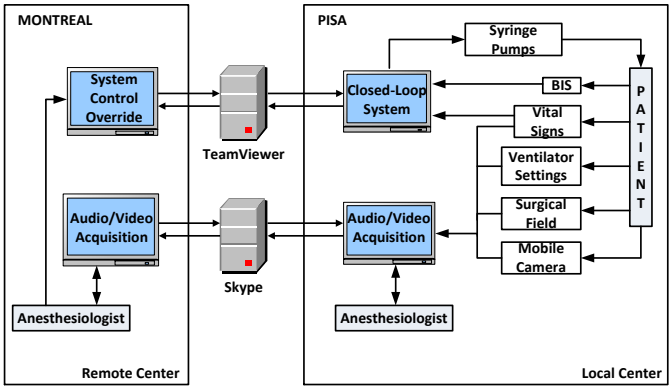


Figure 4–6: Diagram showing the transcontinental anesthesia set-up and operation.

After the interruption of the drug infusion at the end of surgery, when BIS was above 60 for more than 30 consecutive seconds, the system would prompt the anesthesiologist via a voice command to wake-up and extubate the patient.

Finally, at any time, the system can be easily overridden by the user, with a touch of a button, if the anesthesiologist feels there is a need to take control of the anesthesia.

4.2.4 Telemedical Application

This system can be controlled from any PC, smartphone or tablet computer with Internet access, allowing an anesthesiologist to monitor and control the patient's general anesthesia. This system could also be deployed in remote areas of the world where there is a shortage of qualified anesthesia providers, allowing an anesthesiologist to oversee patient care from a distance. One such system was tested between two hospital centers (Montreal General Hospital, Canada and Cisanello Hospital, University of Pisa, Italy) (Hemmerling et al., 2011:A36). The set-up is illustrated in Fig. 4–6.

4.3 Clinical Protocol

After Institutional Ethics Committee approval (McGill University Health Center, Montreal General Hospital, Montreal, QC) and written informed consent, 15 patients undergoing elective surgery requiring general anesthesia were enrolled in the protocol (Hemmerling & Charabti, 2009:A460).

The clinical performance for hypnosis was defined as the efficacy to maintain BIS as close to the target as possible. Relative to the target value of 45, four categories were defined: excellent, good, poor and inadequate control for BIS values within 10%, between 11% and 20%, between 21% and 30% and greater than 30% of the target BIS, respectively.

The clinical performance of analgesia was defined as the efficacy to maintain the AS as close to the target as possible. Four categories were also defined: excellent pain control for AS values between -3 and +3, good pain control for AS values from -6 to -3 and from +3 to +6, and insufficient pain control for AS values from -9 to -6 and from +6 to +9. The fourth category called "other" refers to periods where no score was determined when vagal reactions (decrease of HR solely, without significant change in MAP) and hypovolemia (increase in HR solely) occured.

The precision of the system was assessed using Varvel's performance indices (Varvel et al., 1992), which was conceived originally to evaluate the predictive performance of computer-controlled infusion, but is widely used to assess the performance of closed-loop systems (Struys et al., 2001; G. Puri et al., 2007; Locher et al., 2004). The Varvel paramaters are: performance error (PE) defined as the difference between the real

Table 4–1: Varvel Performance Indices

Index	BIS	MAP	HR
MDPE (%)	-5.9±4.4	8.3±6.9	0.7±6.3
MDAPE (%)	10.3±2.8	10.5±4.9	6.2±3.6
Wobble (%)	8.4±2.1	8.3±2.1	4.9±2.2
Divergence (%/min)	-0.024 ±0.057	-0.048 ±0.099	-0.029 ±0.049

and target values, median performance error (MDPE) is the measure of bias to describe the direction of the error, median absolute performance error (MDAPE) that indicates the size of the error, wobble is a measure of the intra-individual variability in PE, and divergence that reflects the evolution of the controller's performance through time (slope of the regression curve of absolute PE over time).

4.4 Results

Patients (3 F, 12 M; age 57±13 y; weight 79±16 kg) underwent anesthesia for a duration of 191±66 min. Patients received mean doses of propofol of 129±23 μg/kg/min, remifentanil of 0.16±0.06 μg/kg/min and rocuronium of 1.5±0.48 mg/kg. The Varvel performance indices for this system are presented in Table 4–1. The clinical performance of hypnosis is shown in Fig. 4–7, where excellent and good control of hypnosis were obtained during 50% and 33% of the anesthesia duration, respectively. The clinical performance of analgesia is shown in Fig. 4–8, where the score was excellent and good during 69% and 25% of the anesthesia duration, respectively. No score could be determined during 5% of the time.

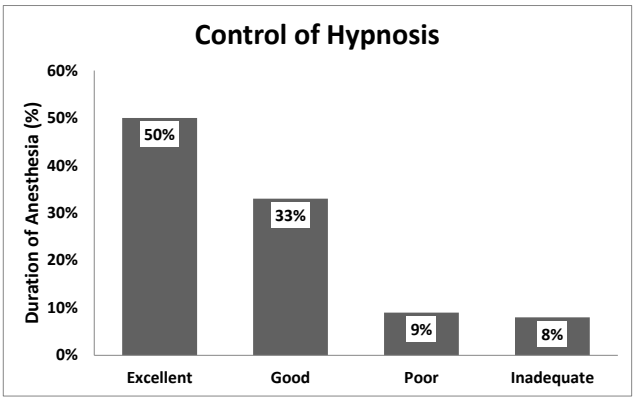


Figure 4–7: Control of hypnosis.

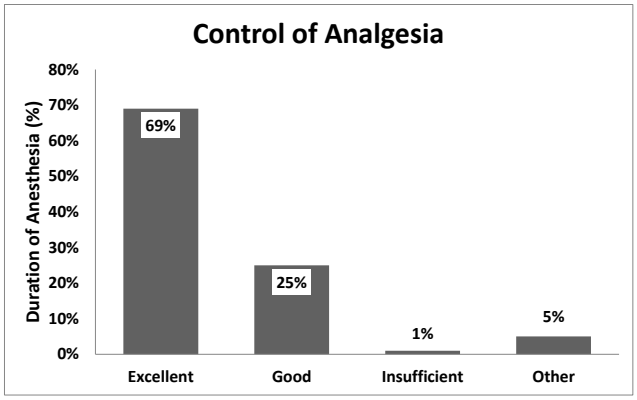


Figure 4–8: Control of analgesia.

4.5 Discussion and Conclusion

We present the first completely automated anesthesia delivery system of all three components of general anesthesia, used during induction, maintenance and emergence from general anesthesia. McSleepy can be viewed as a pharmacological robot, which combines a PID-controller and self-adaptive algorithms controlling a feedback system and offering an intuitive graphical user interface on a large touch screen. It can also be regarded as an anesthesia management information system

since it records drug dosing, vital sign monitoring and surgery-related parameters, such as the time of induction, disinfection, positioning, incision or emergence from anesthesia. It uniquely communicates with the user via voice indicating specific actions which are important for anesthesiologists, such as when specific drugs are given, it indicates the necessity to perform certain acts, such as face mask ventilation, intubation or wake-up from anesthesia based on objective parameters, e.g. certain levels of consciousness. It also offers telemedical capabilities through its video feeds.

Puri *et al.* (G. Puri et al., 2007) have used a PID controller in a closed-loop anesthesia delivery system (CLADS) to control propofol hypnosis during general anesthesia in a wide range of surgeries and patients. They found that the automated control of hypnosis provided better control of BIS with minimal fluctuation, a more economical drug usage and less periods of excessive anesthetic depth than the manual control. However, CLADS only controls one component of general anesthesia, hypnosis, using a "black box" type system without offering any of the additional features of McSleepy, such as voice communication or the advanced user interface.

Liu *et al.* (Liu et al., 2011) have presented a closed-loop delivery system for remifentanil and propofol for analgesia and hypnosis, which they called a "dual" system since it controls two drugs. However, it uses only one controlling variable, the BIS, based on the theory that variations in the BIS also indicate levels of nociception. However, only a few studies have validated this theory (Iselin-Chaves et al., 1998; Guignard et al., 2000). Similar to CLADS, a computer is used without an extensive user interface or the additional features offered by McSleepy. The dual system has been used in a wide range of surgeries and patient population, including transplant

surgery (Liu et al., 2008).

The assessment of pain during general anesthesia is not an easy task. Communication with the patient is impossible, thus indirect parameters must be used. Carregal *et al.* (Carregal et al., 2000) proposed a system to control analgesia using HR and MAP as the control variables. The Analgoscore represents a measure of perioperative nociception as indicated by changes in hemodynamic variables such as the HR and MAP (Hemmerling et al., 2009). This score of nociception successfully controls the closed-loop infusion of remifentanyl for the control analgesia during general anesthesia.

Neither CLADS nor the dual system offer any of the safety features built into McSleepy, lockout time for administration of muscle relaxants 20 min before the end of surgery or the various safety features during induction discussed in section 4.2.3. Voice clips indicating what McSleepy is doing can, specifically during induction, help the anesthesiologist to concentrate on other tasks, such as hemodynamic control or control of breathing/ventilation. The fact that definite time points of surgery are indicated by the user to McSleepy, such as positioning or disinfection, and are integrated in the control algorithm are also unique and have the potential to improve the control mechanism. Anesthesiologists like to treat pain before it arrives; they would therefore typically increase the dose of remifentanil before incision of surgery in order to deliver sufficient analgesia at that point. A closed loop system which is not "aware" of the impending incision will only react to it after the noxious stimulus has occurred. By indicating the time points during surgery to McSleepy, the device can act before noxious stimuli occur or adjust dosages of drugs according to the

progress of surgery.

The integration of live feeds from external cameras and the capability for voice control as well as the possibility for remote desktop control from any handheld device integrate various tele-anesthesia capabilities. These have been used to perform the first transcontinental tele-anesthesia experiment (Hemmerling et al., 2011:A36); further studies are needed to explore its full telemedical capacity.

In conclusion, the first completely automated anesthesia delivery system, controlling hypnosis, analgesia and neuromuscular blockade has been presented. The system will be used as a platform to integrate more features, such as voice command and fluid management. A large scale clinical trial is undertaken to compare the performance of this system versus manually controlled anesthesia delivery.

CHAPTER 5
Article 3

Robot-assisted endotracheal intubations versus endotracheal intubations using a videolaryngoscope: comparison of learning curves using a novel difficult airway mannequin

Mohamad Wehbe[*], Joshua Morse[*], Shantale Cyr[‡], Christophe Philippona[*], Marilu Giacalone[†], Thomas M Hemmerling[*‡#]

[*]Department of Experimental Surgery, McGill University, Montreal, Canada

[‡]Department of Anesthesia, McGill University, Montreal, QC, Canada

[†]Department of Anesthesiology, University of Pisa, Pisa, Italy

[#]Arnold and Blema Steinberg Simulation Centre, McGill University, Montreal, QC, Canada

Contribution of Authors

- Mohamad Wehbe: study setup, data analysis, and preparation of the first draft of the manuscript.
- Joshua Morse: device setup.
- Shantale Cyr: revision of manuscript.
- Christophe Philippona: study setup and data management.
- Thomas Hemmerling: data analysis, revision of manuscript versions, IP.

Summary

Background: Endotracheal intubation is a common procedure of general anesthesia in order to provide artificial ventilation. We developed a robot-assisted intubation system, the Kepler Intubation System, that allows for robot-assisted tracheal intubations. The objective of this study is to compare the performance of the Kepler Intubation System on a standard airway mannequin with a difficult airway mannequin. The hypothesis of this study is that robot assistance will result in a faster skill acquisition than manual procedures for intubations.

Methods: In this study 2 airway mannequins were used, a standard airway mannequin and an adjustable airway mannequin on two settings: standard and difficult. The groups are as follows: robot-assisted on difficult setup, manual on difficult setup, robot-assisted on standard setup, manual on standard setup, robot-assisted on standard mannequin, manual on standard mannequin, and semi-automated on difficult setup. Twenty intubation trials were done per group. Intubation times were measured for each trial; from the moment the blade of the videolaryngoscope entered the mouth of the airway mannequin, to the moment of insertion of the endotracheal tube in the mannequin's trachea. Semi-automated intubations were also performed on the adjustable difficulty mannequin on difficult setting. Performance times were compared using the Student's t-test. $P < 0.05$ was considered significant. Data presented as mean (standard deviation).

Results: All intubation attempts were successful at first attempt. Linear regression showed that the decrease in time between consecutive robot-assisted trials was significantly ($P = 0.04$) greater than the decrease for manual intubations.

Conclusion: Robot-assisted endotracheal intubations allow for faster skill acquisition over their manual counterparts.

Word count: 250

5.1 Introduction

Robot assistance in surgery has been shown to provide an increased precision of movements, improve patients outcomes (Willis et al., 2012; Lau et al., 2012), and reduce morbidity (Ramsay et al., 2012). Also, studies have shown that novices can acquire robot assistance skills in surgery with relative ease (Brinkman et al., 2013; Hanly et al., 2004). Additionally, robot assistance in surgery has been shown to help achieve shorter learning curves and better accuracy than manual procedures (Heemskerk et al., 2007; Rashid et al., 2010).

In fact, a study by Hanly and colleagues in 2004 showed that operative times for laparoscopic surgeries decreased by 39% by the third practice operation (Hanly et al., 2004). Another study by Brinkman and colleagues in 2013 showed that more than half of the participants achieved an expert-level proficiency in robot-assistance in surgery after only 10 trials (Brinkman et al., 2013).

In anaesthesia however, little research has been done on robot assistance. Only very recently, attempts have been made to create robot assistance in anaesthesia. Tighe's group (Tighe, Badiyan, Luria, Lampotang, & Parekattil, 2010) has used the da Vinci Surgical System (Intuitive Surgical, Inc., Sunnyvale, CA, USA) recently to perform two fibre optic intubations.

As several studies have indicated that a steeper learning curve can be obtained for robot-assisted versus manual surgery (Brinkman et al., 2013; Hanly et al., 2004;

Heemskerk et al., 2007; Rashid et al., 2010), a similar trend could be expected for robot assistance in anaesthesia. In this article, we present the first specifically designed anaesthetic robotic system, called Kepler Intubation System (KIS), to perform routine videolaryngoscopic intubations. The Kepler Intubation System is used to perform robot-assisted tracheal intubations via a videolaryngoscope mounted on a robotic arm. The user controls the robotic arm through the use of a joystick.

The Kepler Intubation System was used in this study on a standard airway mannequin, and on an adjustable difficulty airway mannequin in order to simulate a difficult intubation.

In this study, we also compare the success rates, learning curves, and performance times of robot-assisted versus manual intubations on the standard and adjustable difficulty airway mannequins. The hypothesis of this study is that robot assistance will result in faster intubation skill acquisition than manual intubations.

5.2 Methods

The Kepler Intubation System is designed to allow anesthesiologists to perform safe, robot-assisted endotracheal intubations of anaesthetized patients. The system is composed of a video laryngoscope (Pentax AWS, Ambu A/S, Ballerup, Denmark) mounted on a robotic arm (JACO robotic arm, Kinova Rehab, Montreal, QC, Canada) that is controlled via a software control centre and joystick. The graphical user interface for the system features a view form the video feed of the video laryngoscope and a lateral camera view of the intubation procedure (Figure 5–1).

A cart designed specifically to mount the robotic arm and hold the computer and joystick is placed 42 cm from the mannequin's head (a distance that was experimentally

determined as the ideal distance for intubation to start). The video laryngoscope is mounted to the robotic arm using a custom clamp and the webcam is placed to provide a lateral view of the patient's mouth.

The anaesthesiologist maneuvers the robotic arm to place the video laryngoscope above the mouth of the mannequin. Recording the intubation time begins when the anaesthesiologist starts maneuvering the tip of the video laryngoscope into the mouth. Using the lateral view of the webcam and the video feed from the laryngoscope, the anesthesiologist then moves the tip of the blade to the back of the throat, around the tongue and into the pharynx until the epiglottis is visible. Once the vocal cords are located and centered on green targeting crosshairs provided on the video feed, the endotracheal tube is glided down its track and into the trachea (left side of Figure 5–1). Once the endotracheal tube is inserted into the trachea the time recording stops.

The control scheme of the KIS is depicted in Figure 5–2. Each of the six degrees of freedom of the robotic arm is mapped to a button or axis of the joystick. The six degrees of freedom allow for the system to mimic all of the possible movements of the human wrist: flexion, extension, radial and ulnar deviation, supination, and pronation. The trigger of the joystick is used to mark the beginning of the intubation procedure. Additionally, a button is mapped to begin and stop recording movements of the system. These recorded movements can be replayed using the graphical user interface (GUI). Two buttons on the top of the joystick are used to either increase or decrease the speed of the robotic arm.

In this study, an anaesthesiologist performed seven different endotracheal intubation

procedures on two different airway mannequins. The airway mannequins used are a standard airway mannequin (Laerdal Airway Management Trainer, Laerdal Medical, Stavanger, Norway), and an adjustable difficulty airway mannequin (Delson et al., 2012) used on standard and difficult setups.

The model we used to simulate a difficult intubation scenario, as shown in Figure 5–3, is a parametrically adjustable intubation mannequin developed by Delson and colleagues (Delson et al., 2012). This model incorporates separate parts for the skull, the face and maxilla, the mandible, the upper and lower teeth, and the cervical vertebrae from C1 to C7 (Delson et al., 2012). This model allows adjustments in the height of the maxilla, the length of the upper incisors, the tension and distance of the jaw movement representing the mouth opening, the stiffness of the spine, the anterior-posterior displacement of the jaw relative to the maxilla and skull, the presence or absence of the upper and lower teeth, the height of the lower jaw, and the height of the headrest. All of these anatomical variations can affect the difficulty of the intubation procedure and are illustrated in Figure 5–3. A comparison of this model to a published well-proportioned model skull was made and it was found that the distances between bone and soft-tissue landmarks of the adjustable model were within a standard deviation of literature reported values (Delson et al., 2012).

The intubations were separated into seven groups as follows (Figure 5–4): manual intubation on the standard mannequin, manual intubation on the adjustable difficulty mannequin (ADM) on standard setup, manual intubation on the ADM on difficult setup, robot-assisted on the standard mannequin, robot-assisted on the ADM on standard setup, robot-assisted on the ADM on difficult setup, and semi-automated

on the ADM on difficult setup.

In the context of this study, a manual intubation is defined as performing a standard endotracheal intubation using the videolaryngoscope.

In every group, one anaesthesiologist performed twenty intubations. In the semi-automated group, the anaesthesiologist performed one robot-assisted intubation, where the intubation was recorded, and then it was replayed for twenty trials.

Safety Features:

A medical robot can either be fail-safe (i.e., enter a safe state upon any system error) or fault-tolerant (i.e., the system continues to operate even in the presence of errors) (Kazanzides, 2009). This system is designed to be fail-safe, as any failure of the system will cause it to enter a state that presents no risks to the patient. If any component of the system is disconnected from another (e.g., if the USB cable connecting the computer and the robotic arm is unplugged), the robotic arm will immediately cease all movement and the video laryngoscope can be removed manually, with no danger to the patient. Both the control computer and robotic arm are plugged into an uninterruptible power supply (Back UPS XS 1300, APC, W. Kingston, RI, USA) with sufficient battery capacity to allow the intubation procedure to be reversed in the case of a power outage.

An important safety feature of the KIS is the movement limitations that are enabled when the trigger of the joystick is clicked to engage intubation mode. These limitations prevent the robotic arm from moving outside of the narrow range of movement necessary to intubate a patient. The robotic arm will also shut down when a spike of current occurs in the motor due to too great a resistive force on the motors: this

will lock the robotic arm in place if it comes into contact with the patient's tissue. The robotic arm has six independently-controlled motors which provide 6 degrees of freedom and are fully back-drivable when shutdown: this allows a safe and easy manual reversal of the intubation procedure for any reason, such as when power is cut to the arm.

Risk Analyses:

In order to evaluate the potential problems that could occur during the use of the Kepler Intubation System and the effect these issues would have on patient safety, a risk analysis was performed. The risk analysis consisted of a failure mode effects analysis (FMEA). The FMEA is a procedure that examines basic potential defects or failures in the components of a system, assesses their effect, and identifies methods of addressing them. Table 5–1 lists the FMEA table for the KIS.

In addition to the component errors listed above, user error is another serious source of error. The greatest risk comes from either the possibility of controller or user error. In order to address these risks, the safety limits described in this paper were implemented. The loss of video signal is a potential problem without a built-in control. In the event of the loss of the video, the arm can be easily powered down and manually removed from the patient's mouth and then intubation can proceed in a standard fashion.

5.3 Results

All intubation trials were successful at first attempt. An intubation was considered successful upon visualization of the vocal cords, and insertion of the tube in

the trachea. Intubation times for all procedures were measured from the beginning of the procedure (videolaryngoscope tip located above the mannequin's mouth) to the successful insertion of the tube in the trachea.

The mean times to perform the manual intubations in the standard mannequin, adjustable difficulty mannequin on standard setup, and adjustable difficulty mannequin on difficult setup are 4.40 (1.27), 6.30 (7.65), and 8.60 (4.84) s, respectively.

The mean times to perform the robot-assisted intubations in the standard mannequin, adjustable difficulty mannequin on standard setup, and the adjustable difficulty mannequin on difficult setup are 32.75 (4.71), 48.95 (21.43), and 42.20 (16.22) s, respectively. Figure 5–5 shows the times of intubation per trial for robot-assisted, and manual, performed on the adjustable difficulty mannequin on both standard and difficult setup.

Using linear regression for procedures done on the adjustable difficulty mannequin, it was found that the mean time required to perform intubations using the Kepler Intubation System decreased with each successive attempt for 1.2 seconds on the standard setting and 1.6 seconds on the difficult setting.

The average decrease in time for all trials for robot-assisted intubations was significantly ($P = 0.04$) greater than the decrease seen for the manual procedures at 1.4 (0.28) s versus 0.7 (0.02) s (Figure 5–6).

For the adjustable difficulty mannequin, it was found that the manual procedures were significantly faster on the standard setup than the difficult setup ($P = 0.04$). On the other hand, it was found the robotic procedures were significantly faster on the difficult setup than the standard setup ($P = 0.04$).

There was no significant difference between the intubation times of the manual procedures on the standard mannequin and the manual procedures on the standard setup on the adjustable difficulty mannequin. On the other hand, there was a significant difference between the intubation times of the robotic procedures on the standard mannequin and both the standard setup (P = 0.002) and the difficult setup (P = 0.002) on the adjustable difficulty mannequin.

The semi-automated group (Figure 5–7), performed on the adjustable difficulty mannequin, had a significantly lower standard deviation than the manual and robotic groups (P = 0.02).

5.4 Discussion

This study demonstrates that the use of robot assistance in endotracheal intubation can result in faster intubation skill acquisition. The first ventures into robot assistance in anesthesia were done by Tighe and colleagues using the da Vinci Surgical System to perform a nerve block in simulation on an US nerve phantom and a simulated fibre optic intubation on an airway trainer mannequin (Tighe, Badiyan, Luria, Lampotang, & Parekattil, 2010; Delson et al., 2012; Kazanzides, 2009; Hemmerling et al., 2012; Tighe, Badiyan, Luria, Boezaart, & Parekattil, 2010). Afterwards, specific robot assistance systems were developed in our lab both for peripheral nerve blocks (Hemmerling et al., 2012; Morse et al., 2013) and endotracheal intubations (Hemmerling et al., 2012).

Difficult intubations are intubations that require multiple attempts in the presence or absence of tracheal pathology (American Society of Anesthesiologists Task Force

on Management of the Difficult Airway, 2003). Some anthropometric factors influence the degree of difficulty of the maneuver: the degree of mouth opening, tongue dimensions, the ability of jaw subluxation, and neck mobility. The adjustable airway model we used provides adjustments to these parameters (Delson et al., 2012). This model could mimic what is encountered in clinical practice.

Results obtained from this study showed that although the robot-assisted intubations took longer to perform than manual intubations, they provided a shorter learning curve. These findings mirror the findings in studies comparing laparoscopic surgery to robot-assisted prostatectomies, where it was shown that for surgical novices the learning curve was shorter using robot assistance (Rashid et al., 2010). Also, similar to the trend where robot-assisted prostatectomies take longer time than the manual procedures (Menon et al., 2002; Tomaszewski et al., 2012), robot-assisted intubations took longer time than manual intubations in this study.

In previous studies comparing robot-assisted procedures to manual procedures, authors have used the average time to complete a certain task by the user to determine the ease of skill acquisition and comparing learning curves between robot-assisted and manual procedures (Hanly et al., 2004; Yohannes et al., 2002). Yohannes and colleagues compared average times to complete certain tasks, such as suturing, from the first and last trials for all participants, using the da Vinci surgical robot and manually, to determine that participants showed faster improvement using robot-assistance (Yohannes et al., 2002). In our study, since we had once participant performing the intubations (manual, and robot-assisted), we used linear regression

by fitting the times to complete every trial to a straight line (as shown in Figure 5–5), and used the slope of the lines to compare the speed of skill acquisition between manual and robot assisted.

In the present study, it was shown that using the robot-assisted technique; intubations on the difficult setup of the adjustable airway mannequin were faster on average than on the standard setup. This could mean that the movements made using the joystick and robotic arm are better suited for difficult intubation scenarios than standard ones.

The learning curves in this study using the adjustable airway mannequin on standard and difficult setup are similar to learning curves obtained from a previous study using a standard airway mannequin (Hemmerling et al., 2012).

Using the semi-automated procedure, the slope of the linear regression line was zero and had a low standard deviation of 0.5 s, demonstrating the high degree of repeatability of an automated intubation approach. This result mirrors a similar outcome seen on a standard airway mannequin in a previous study (Hemmerling et al., 2012). This study demonstrates that the use of robot assistance for endotracheal intubations could reduce the time to acquire the skill compared to the manual procedures. Also, difficult intubations could be performed with less complications using robot assistance.

Another important impact of using robot assistance for performing endotracheal intubations may be the increase of successful intubations done by out-of-hospital rescuers: studies have shown that conventionally-trained out-of-hospital rescuers often fail to accomplish endotracheal intubation in patient requiring invasive airway

management (Wang et al., 2003, 2001). It could be argued that the use of robot assistance in semi-automated mode to perform endotracheal intubation would increase the success when used by out-of-hospital rescuers. Additional studies could be performed to further test this.

There are several limitations to this study. This study was conducted using a single videolaryngoscope and by one operator: the use of different types of videolaryngoscopes and the inclusion of more operators would yield a better dataset for analysis.

In conclusion, in this study we showed that the use of robot assistance allows for faster skill acquisition of endotracheal intubation versus traditional, manually performed intubations. We also showed that semi-automated intubations are feasible and demonstrate repeatability.

Tables

Table 5–1: Failure mode effect analysis table for the Kepler intubation system

Component	Failure Mode	Effect	Detection Method	Applicable Controls
Robotic arm and computer	Loss of Power	The robotic arm will stop moving	The battery unit will sound an alarm	Battery backup power is provided for the computer and robotic arm
Computer	Software crash	The system will stop transmitting commands to the robotic arm	The computer screen will show a crash report	The robotic arm will stop movement
Controller	Error	The controller will transmit unintended messages to the arm	The arm will move unexpectedly	Safety limits are in place to limit the scope of harm to the patient
Video	Loss of signal	The video feed from the laryngoscope will be unavailable	The video feed will turn black	None

Figures

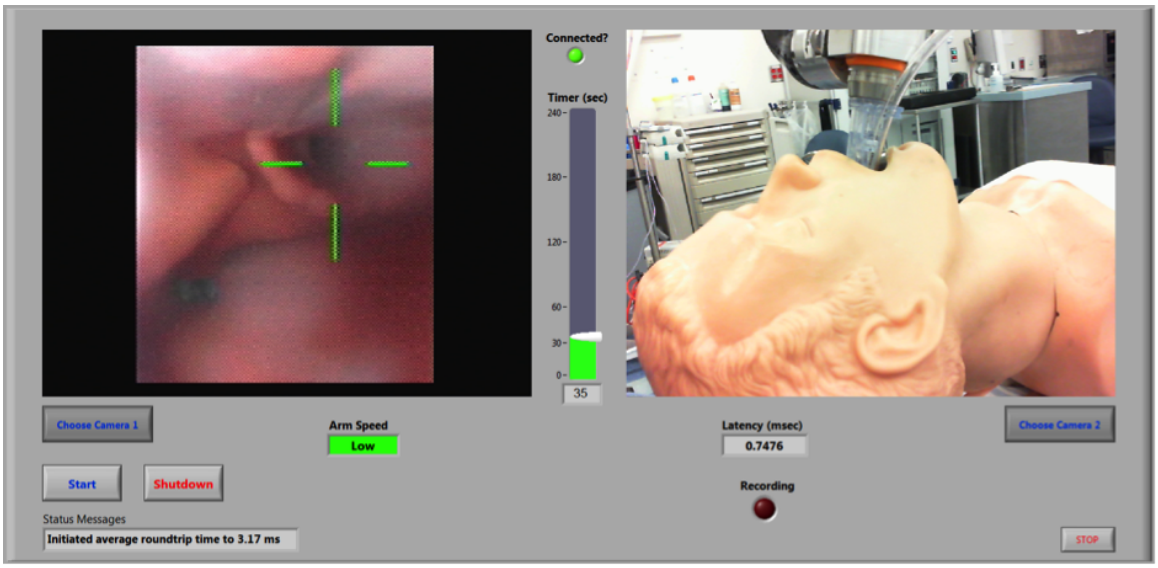


Figure 5–1: The graphical user interface of the Kepler Intubation System. Left: Video from laryngoscope. Right: Lateral view of video from webcam. Between the video feeds is the intubation timer. Below the video feeds are the arm speed and network latency displays.

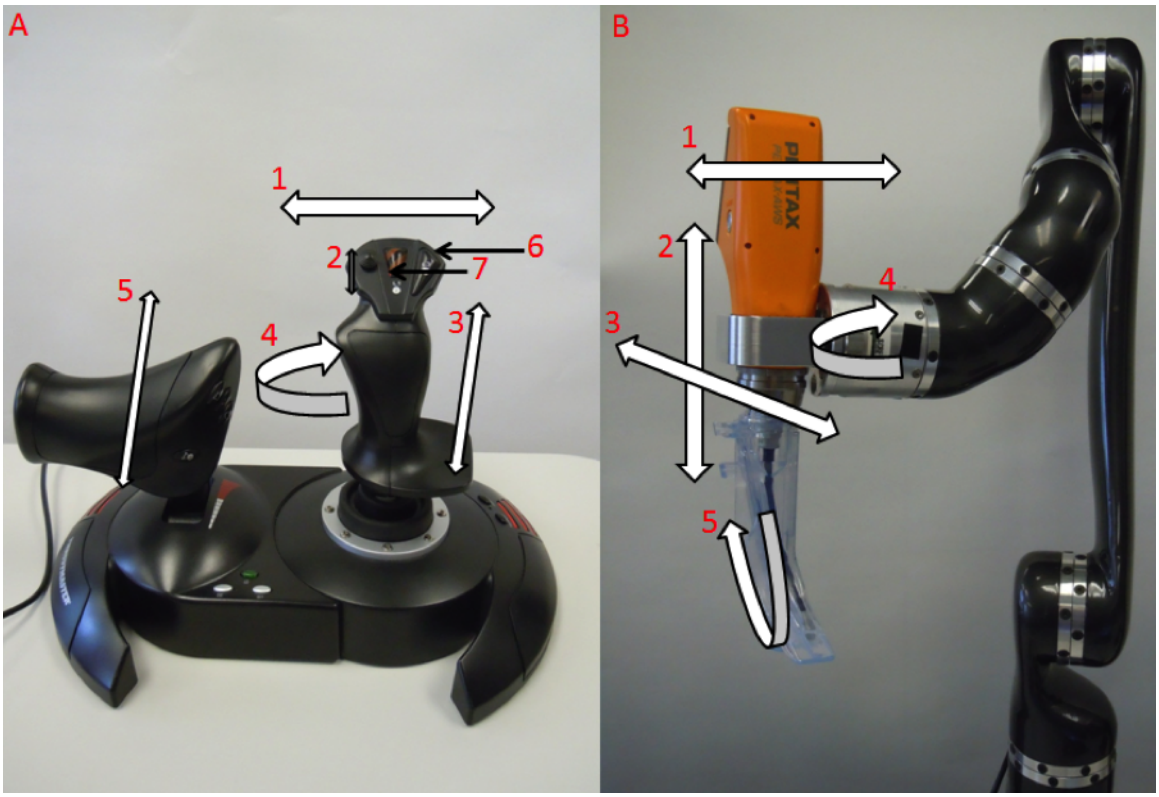


Figure 5–2: Control scheme of the Kepler Intubation System. (A) Movements of the video laryngoscope; (B) corresponding movements of the joystick. 1, left and right; 2, up and down; 3, forward and backward; 4, rotation of the blade horizontally; 5, rotation of the tip of the video laryngoscope; 6, decreasing the speed; 7, increasing the speed.

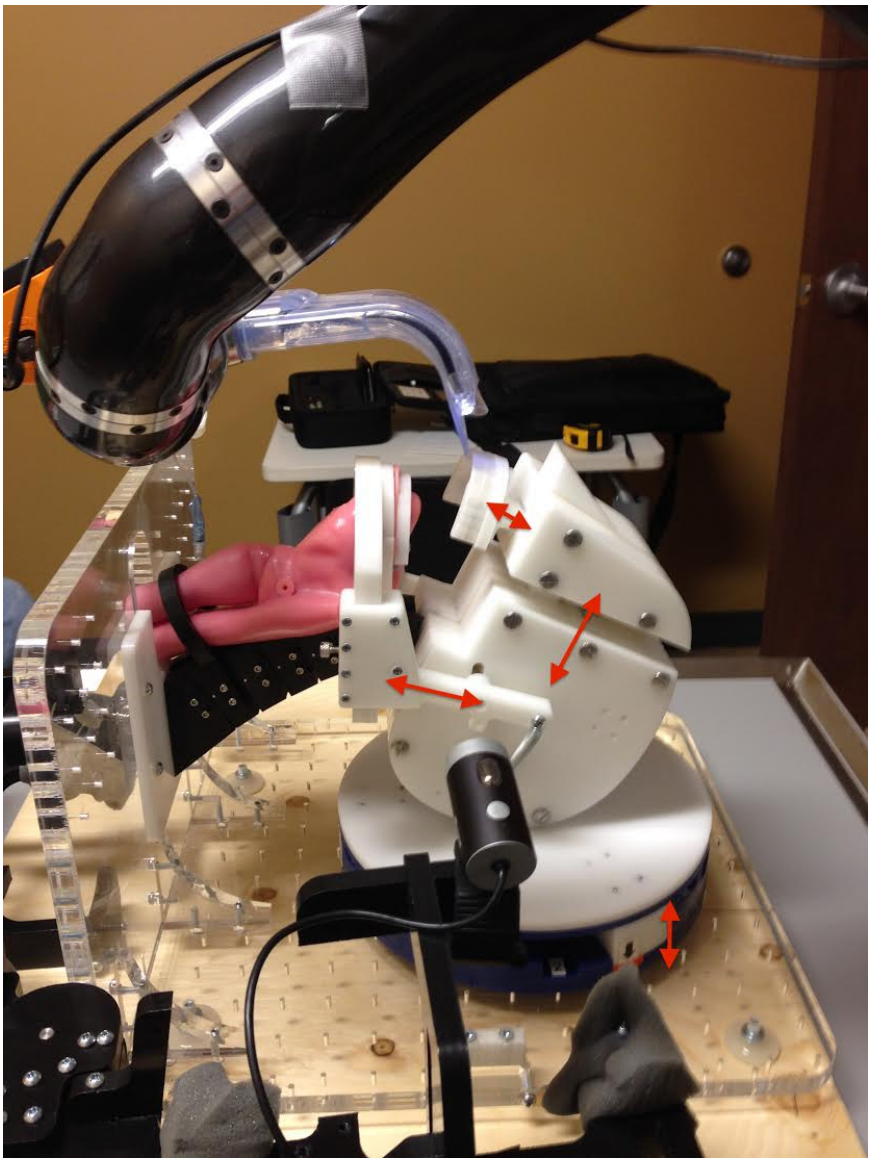


Figure 5–3: The setup of the robot-assisted intubation using the adjustable difficulty airway mannequin. The arrows represent the direction of the adjustment that can be applied to the airway mannequin in order to adjust the difficulty of the intubation.

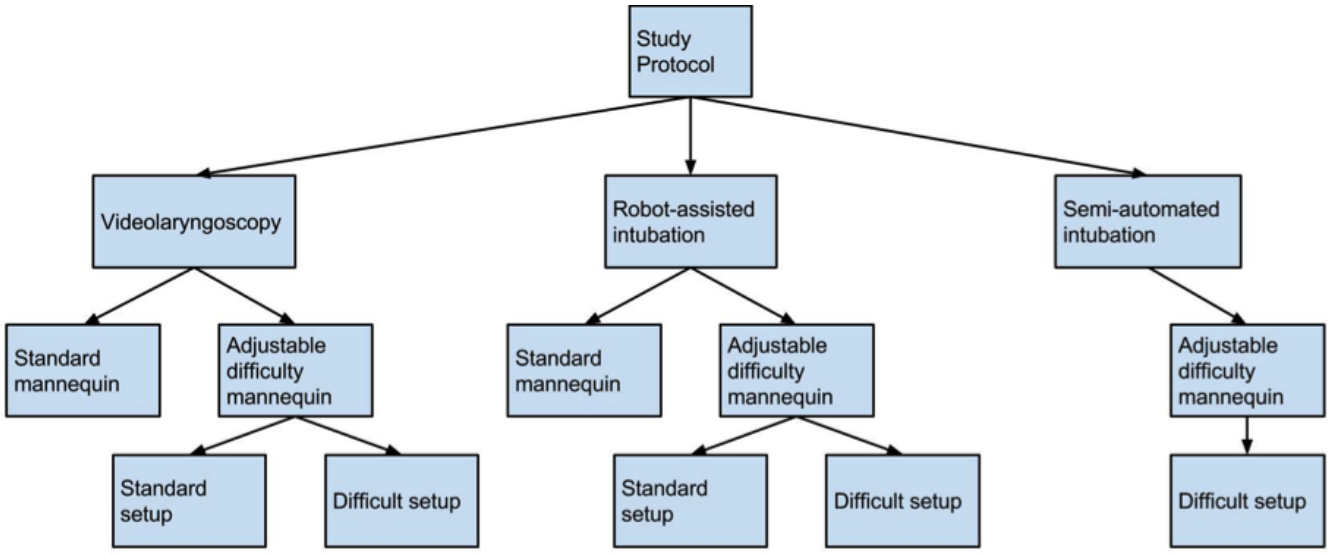
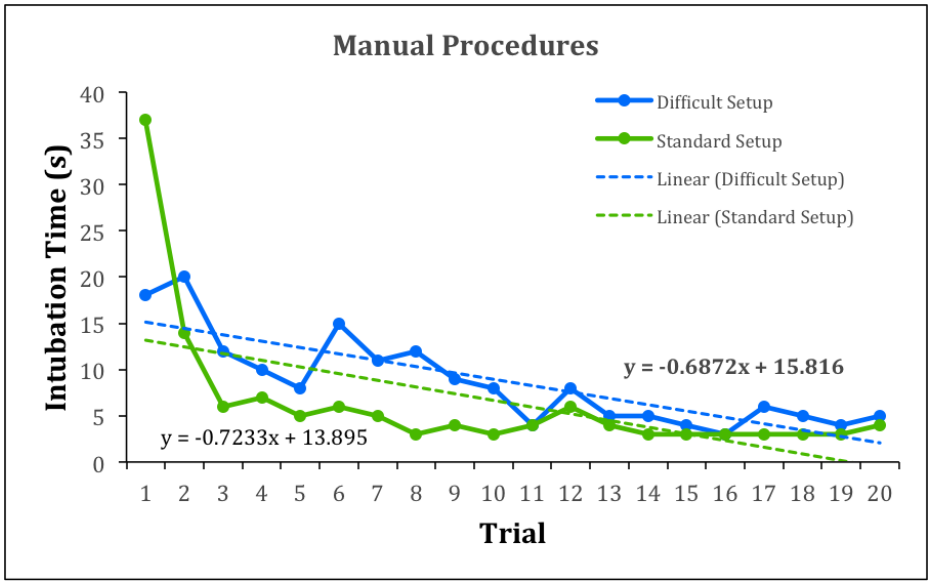
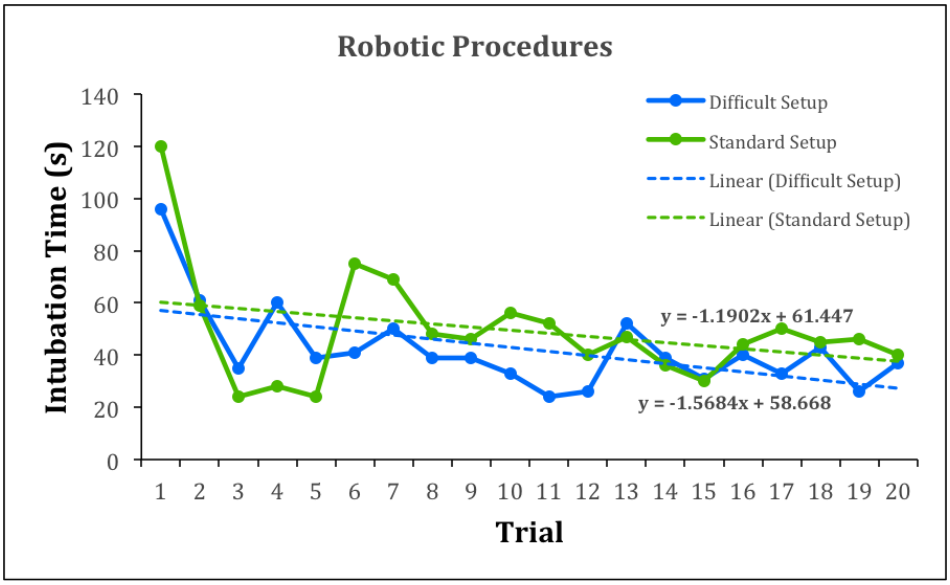


Figure 5–4: Flowchart illustrating the study setup.



(a)



(b)

Figure 5–5: Intubation times for manual (top) and robotic (bottom) procedures on the adjustable mannequin in standard and difficult setup.

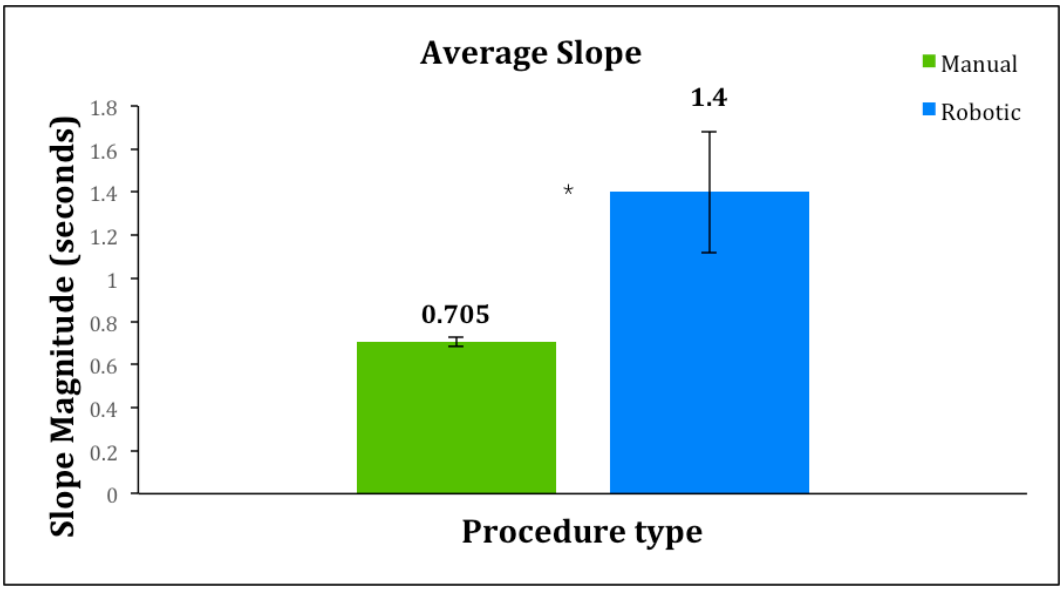


Figure 5–6: Magnitudes of the slopes of the linear regression lines for manual and robot assisted intubations on both the standard and difficult settings of the adjustable difficulty mannequin. *: P = 0.04.

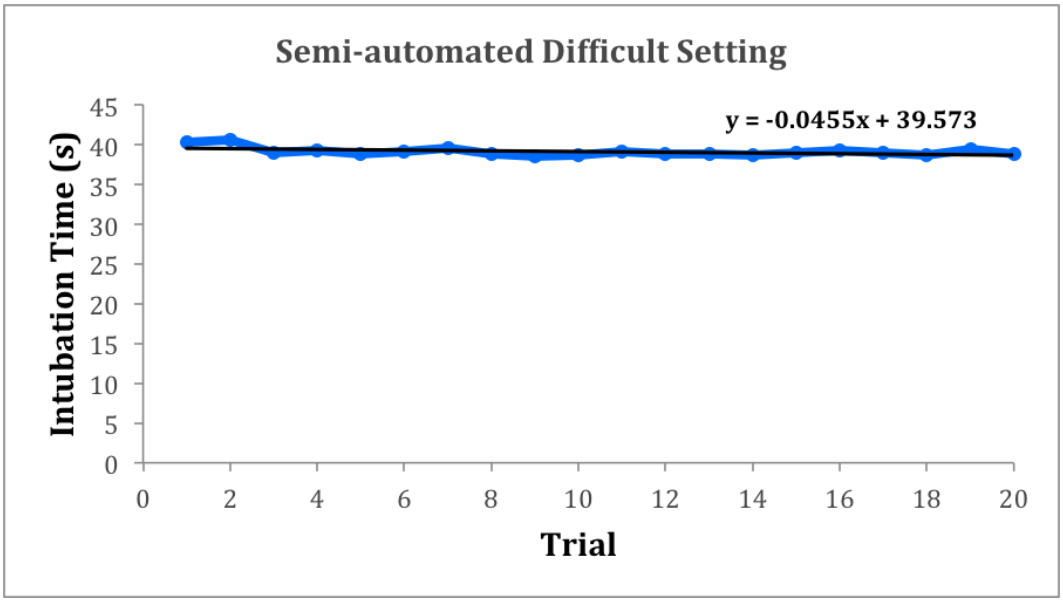


Figure 5–7: Intubation times for the semi-automated procedures on the adjustable mannequin in difficult setup.

CHAPTER 6
Discussion and conclusion

The idea behind this project was to explore the role of robotization and automation in anesthesia. More specifically, it was based on the hypothesis that it is possible to automate all three components of general anesthesia (i.e., hypnosis, analgesia, and neuromuscular blockade) at all stages of a surgery, from induction, to maintenance, and finally to emergence. Additionally, this project was based on the hypothesis that robot-assisted endotracheal intubations allow for faster skill acquisition than their manual counterparts. This work aims to reduce the clinician's workload during surgery in order to reduce medical errors and iatrogenic complications, and to improve the skill acquisition of intubation techniques by integrating robot assistance.

The primary results demonstrate that closed-loop systems allow for autonomous control of general anesthesia, while providing a superior performance when compared to the manual administration of anesthetic agents.

The presented system uses TIVA in a closed-loop fashion to deliver general anesthesia. Both TIVA and anesthesia using volatile anesthetics have their advantages and disadvantages. Anesthesia using volatile anesthetics has a lower cost and is less labor intensive. Total intravenous anesthesia, on the other hand, provides improved quality of emergence from anesthesia, reduced postoperative nausea and vomiting, rapid onset of action independent from the alveolar ventilation, and can easily be integrated into closed-loop drug delivery systems.

The performance of the automated delivery system in comparison to human, manual control was tested in a randomized controlled trial (Hemmerling et al., 2013). One-hundred and eighty-six patients undergoing elective surgery were enrolled and randomized in two groups (closed-loop group, and control group) of 93 patients each. Anesthesia control in the closed-loop group showed better control of hypnosis and analgesia than in the control group. These results are promising and indicate the usefulness of automated anesthesia delivery. Obviously, larger scale patient studies are needed before a routine clinical use can be envisioned.

Our findings also demonstrate that the use of robot assistance can allow for the skills required for endotracheal intubations to be learned faster.

Closed-loop control systems try to maintain a measured output variable at a set point. An important part of a closed-loop system is thus the feedback control variable. In anesthesia, this feedback control variable is frequently a single or a set of monitoring parameters.

Bispectral index has been used in clinical practice for hypnosis monitoring since 1997 (Johansen, 2006). Following the introduction of the bispectral index, research on closed-loop systems in hypnosis has used this variable as a feedback for hypnosis control. Since the reliability of any closed-loop system is not only based on the reliability of the controller, but also the reliability of the feedback parameter, numerous studies have documented the reliability of the bispectral index and a wealth of clinical research has accumulated on its use.

On the other hand, the assessment of pain during general anesthesia isn't as straightforward as hypnosis. Communication with a patient under hypnosis is impossible, so

indirect parameters must be used. Anesthesiologists rely on the patient's hemodynamics (i.e., heart rate and mean arterial blood pressure) to assess the pain during surgery. Several research groups have proposed closed-loop systems based on hemodynamics to control analgesia (Carregal et al., 2000; Hemmerling et al., 2009). It is worth mentioning that one research group proposed the use of the bispectral index to titrate analgesics in a closed-loop (Liu et al., 2011). Their assumption was that painful intra-operative stimuli provoke cortical activation and consequently an increase in the bispectral index.

Puri *et al.* (G. Puri et al., 2007) used a closed-loop anesthesia delivery system to control hypnosis during general anesthesia in a wide variety of surgeries and patients. Not unlike our results, they reported a better control of BIS with more economical drug usage over manual control. Their system, however, controls only one component of anesthesia, hypnosis. Liu *et al.* (Liu et al., 2011) have presented a closed-loop system for delivery of propofol and remifentanil using only bispectral index as a feedback control variable for both components. This system, however, did not have closed-loop control for neuromuscular blockade.

Manual robots, on the other hand, are not controlled through a closed-loop; they are controlled by an operator. In anesthesia, these robots are used to assist clinicians to perform manual gestures, such as intubation or regional anesthesia. Tighe *et al.* (Tighe, Badiyan, Luria, Boezaart, & Parekattil, 2010; Tighe, Badiyan, Luria, Lampotang, & Parekattil, 2010) and Hemmerling *et al.* (Hemmerling et al., 2013b; Hemmerling et al., 2012; Morse et al., 2013) have worked on robots that provide assistance in intubation and regional anesthesia.

This work has several limitations. For example, it was not tested on a wide variety of surgery types nor patient populations. Another limitation is that the closed-loop anesthesia delivery system study was not a double-blinded study, since the anesthesiologist in charge was aware of the group assignment. Another limitation is the lack of a reliable measure for intraoperative pain.

Also, the robot-assisted intubation for skill acquisition was done on a small sample size.

As for future directions, McSleepy should be tested with different types of surgery and patient populations. For possible commercialization, the system needs to be adapted for a real-time operating system that is compliant with IEC 62304, a standard for "Medical device software - Software life cycle processes". To bring this system to market, the "Design & Development" and "Verification & Validation" phases of the product development process have to be reworked.

This system would be considered a class III medical device (Patek et al., 2009). Contemporary computing technology and using *in silico* testing could accelerate the process of medical device development towards a commercial product (Patek et al., 2009).

In silico testing of such systems would be based on software models describing pharmacokinetics and pharmacodynamics of propofol, remifentanil, and rocuronium. The complexity of this software is dependent on the complexity of the mathematical models of the pharmacokinetics and pharmacodynamics of the aforementioned drugs.

The robot-assisted intubation research should be performed on a larger scale, with junior anesthesiologists and residents in anesthesia.

Finally, a distinction should be made regarding the future of both anesthesia systems. Whereas McSleepy was designed to assist anesthesiologists, the KIS, in a completely autonomous mode, could be used in areas where immediate intubation is needed, and where medical experts are not available (e.g., cardiac and respiratory arrest at an airport). The KIS could also be useful in areas where intubations are performed by inexperienced medical personnel, where intubations could take up to 5 minutes or are not successful at all.

References

Absalom, A. R., & Kenny, G. N. (2003). Closed-loop control of propofol anaesthesia using bispectral index: performance assessment in patients receiving computer-controlled propofol and manually controlled remifentanil infusions for minor surgery. *Br J Anaesth*, *90*(6), 737-41.

Absalom, A. R., Sutcliffe, N., & Kenny, G. N. (2002). Closed-loop control of anesthesia using bispectral index: performance assessment in patients undergoing major orthopedic surgery under combined general and regional anesthesia. *Anesthesiology*, *96*(1), 67-73.

Agarwal, J., Puri, G. D., & Mathew, P. J. (2009). Comparison of closed loop vs. manual administration of propofol using the bispectral index in cardiac surgery. *Acta Anaesthesiol Scand*, *53*(3), 390-7.

American Society of Anesthesiologists Task Force on Intraoperative Awareness. (2006). Practice advisory for intraoperative awareness and brain function monitoring: A report by the american society of anesthesiologists task force on intraoperative awareness. *Anesthesiology*, *104*(4), 847-64.

American Society of Anesthesiologists Task Force on Management of the Difficult Airway. (2003). Practice guidelines for management of the difficult airway: an updated report by the american society of anesthesiologists task force on management of the difficult airway. *Anesthesiology*, *98*(5), 1269.

Banerjee, S., Desilets, D., Diehl, D. L., Farraye, F. A., Kaul, V., Kethu, S. R., ... Tierney, W. M. (2011). Computer-assisted personalized sedation. *Gastrointestinal Endoscopy*, *73*(3), 423-427.

Bischoff, P., Schneider, G., & Kochs, E. (2008). Anesthetics drug pharmacodynamics. In *Modern anesthetics* (pp. 379–408). Springer.

Brinkman, W. M., Luursema, J. M., Kengen, B., Schout, B. M., Witjes, J. A., & Bekkers, R. L. (2013). da vinci skills simulator for assessing learning curve and criterion-based training of robotic basic skills. *Urology*, *81*(3), 562-6.

Butterworth, J., Mackey, D., & Wasnick, J. (2013). *Morgan and mikhail's clinical anesthesiology* (5th ed.). Mcgraw-hill.

Carregal, A., Lorenzo, A., Taboada, J. F., & Barreiro, J. L. (2000). [intraoperative control of mean arterial pressure and heart rate with alfentanyl with fuzzy logic]. *Rev Esp Anestesiol Reanim*, *47*(3), 108-13.

Cermack, M. (2006). Monitoring and telemedicine support in remote environments and in human space flight. *British Journal of Anaesthesia*, *97*(1), 107-114.

Charabati, S., Bracco, D., Mathieu, P. A., & Hemmerling, T. M. (2009). Comparison of four different display designs of a novel anaesthetic monitoring system, the 'integrated monitor of anaesthesia (ima^{TM})'. *British Journal of Anaesthesia*, *103*(5), 670-677.

Chu, L., & Fuller, A. (2012). *Manual of clinical anesthesiology* (1st ed.). Lippincott Williams & Wilkins.

Cone, S. W., Gehr, L., Hummel, R., Rafiq, A., Doarn, C. R., & Merrell, R. C. (2004). Case report of remote anesthetic monitoring using telemedicine. *Anesthesia &*

Analgesia, *98*(2), 386-388.

Curatolo, M., Derighetti, M., Petersen-Felix, S., Feigenwinter, P., Fischer, M., & Zbinden, A. M. (1996). Fuzzy logic control of inspired isoflurane and oxygen concentrations using minimal flow anaesthesia. *Br J Anaesth*, *76*(2), 245-50.

de Graaf, P. M., van den Eijkel, G. C., Vullings, H. J., & de Mol, B. A. (1997). A decision-driven design of a decision support system in anesthesia. *Artificial intelligence in medicine*, *11*, 141-53.

Delson, N., Sloan, C., McGee, T., Kedarisetty, S., Yim, W.-W., & Hastings, R. H. (2012). Parametrically adjustable intubation mannequin with real-time visual feedback. *Simulation in Healthcare*, *7*(3), 183-191.

Edwards, N. D., Mason, D. G., & Ross, J. J. (1998). A portable self-learning fuzzy logic control system for muscle relaxation. *Anaesthesia*, *53*(2), 136-9.

Eleveld, D., Proost, J., & Wierda, J. (2005). Evaluation of a closed-loop muscle relaxation control system. *Anesthesia & Analgesia*, *101*(3), 758-64.

Fiadjoe, J., Gurnaney, H., Muralidhar, K., Mohanty, S., Kumar, J., Viswanath, R., ... Rehman, M. (2009). Telemedicine consultation and monitoring for pediatric liver transplant. *Anesthesia & Analgesia*, *108*(4), 1212-1214.

Galvez, J. A., & Rehman, M. A. (2011). Telemedicine in anesthesia: An update. *Current Opinion in Anesthesiology*, *24*(4), 459-462.

Gentilini, A., Morari, M., Bieniok, C., Wymann, R., & Schnider, T. W. (2001). Closed-loop control of analgesia in humans. In *Decision and control, 2001. proceedings of the 40th ieee conference on* (Vol. 1, p. 861-866 vol.1).

Gentilini, A., Schaniel, C., Morari, M., Bieniok, C., Wymann, R., & Schnider, T.

(2002). A new paradigm for the closed-loop intraoperative administration of analgesics in humans. *Biomedical Engineering, IEEE Transactions on*, *49*(4), 289 -299.

Guen, M., Liu, N., Bourgeois, E., Chazot, T., Sessler, D., Rouby, J.-J., & Fischler, M. (2013). Automated sedation outperforms manual administration of propofol and remifentanil in critically ill patients with deep sedation: a randomized phase ii trial. *Intensive Care Medicine*, *39*(3), 454-462.

Guignard, B., Menigaux, C., Dupont, X., Fletcher, D., & Chauvin, M. (2000). The effect of remifentanil on the bispectral index change and hemodynamic responses after orotracheal intubation. *Anesthesia & Analgesia*, *90*(1), 161.

Halford, G., Baker, R., McCredden, J., & Bain, J. (2005). How many variables can humans process? *Psychol Sci*, *16*(1), 70-6.

Hanly, E. J., Marohn, M. R., Bachman, S. L., Talamini, M. A., Hacker, S. O., Howard, R. S., & Schenkman, N. S. (2004). Multiservice laparoscopic surgical training using the davinci surgical system. *Am J Surg*, *187*(2), 309-15.

Heemskerk, J., van Gemert, W. G., de Vries, J., Greve, J., & Bouvy, N. D. (2007). Learning curves of robot-assisted laparoscopic surgery compared with conventional laparoscopic surgery: an experimental study evaluating skill acquisition of robot-assisted laparoscopic tasks compared with conventional laparoscopic tasks in inexperienced users. *Surg Laparosc Endosc Percutan Tech*, *17*(3), 171-4.

Hemmerling, T., Arbeid, E., Tang, L., Cyr, S., Wehbe, M., Giunta, F., & Zaouter, C. (2011:A36). Transcontinental anesthesia. *Anesthesia & Analgesia*.

Hemmerling, T., Arbeid, E., Wehbe, M., Cyr, S., Taddei, R., & Zaouter, C. (2013).

Evaluation of a novel closed-loop total intravenous anaesthesia drug delivery system: a randomized controlled trial. *British journal of anaesthesia*, 1-9.

Hemmerling, T., & Charabti, S. (2009:A460). Mcsleepy – a completely automatic anesthesia delivery system. *Anesthesiology*.

Hemmerling, T., Taddei, R., Wehbe, M., Cyr, S., Zaouter, C., & Morse, J. (2013a). First robotic ultrasound-guided nerve blocks in humans using the magellan system. *Anesthesia & Analgesia*, *116*(2), 491-494.

Hemmerling, T., Taddei, R., Wehbe, M., Cyr, S., Zaouter, C., & Morse, J. (2013b). Technical communication: First robotic ultrasound-guided nerve blocks in humans using the magellan system. *Anesthesia and analgesia*, *116*(2), 491-4.

Hemmerling, T., Wehbe, M., Zaouter, C., Taddei, R., & Morse, J. (2012). The kepler intubation system. *Anesth Analg*, *114*(3), 590-4.

Hemmerling, T. M. (2009). Automated anesthesia. *Curr Opin Anaesthesiol*, *22*(6), 757-63.

Hemmerling, T. M., Arbeid, E., Wehbe, M., Cyr, S., Giunta, F., & Zaouter, C. (2013). Transcontinental anaesthesia: a pilot study. *British journal of anaesthesia*, 1-6.

Hemmerling, T. M., Charabati, S., Salhab, E., Bracco, D., & Mathieu, P. A. (2009). The analgoscore: a novel score to monitor intraoperative nociception and its use for closed-loop application of remifentanil. *Journal of computers*, *4*(4), 311-318.

Hemmerling, T. M., Taddei, R., Wehbe, M., Zaouter, C., Cyr, S., & Morse, J. (2012). First robotic tracheal intubations in humans using the kepler intubation system. *British Journal of Anaesthesia*, *108*(6), 1011-1016.

Himpens, J., Leman, G., & Cadiere, G. B. (1998). Telesurgical laparoscopic cholecystectomy. *Surg Endosc*, *12*(8), 1091.

Iselin-Chaves, I., Flaishon, R., Sebel, P., Howell, S., Gan, T., Sigl, J., ... Glass, P. (1998). The effect of the interaction of propofol and alfentanil on recall, loss of consciousness, and the bispectral index. *Anesthesia & Analgesia*, *87*(4), 949-955.

Janda, M., Simanski, O., Bajorat, J., Pohl, B., Noeldge-Schomburg, G. F., & Hofmockel, R. (2011). Clinical evaluation of a simultaneous closed-loop anaesthesia control system for depth of anaesthesia and neuromuscular blockade*. *Anaesthesia*, *66*(12), 1112-20.

Johansen, J. W. (2006). Update on bispectral index monitoring. *Bailliere's Best Practice in Clinical Anesthesiology*, *20*(1), 81-99.

Johnson, J. . (2013). *Fda grants premarket approval (pma) for the sedasys® system for healthy patients undergoing sedation during routine colonoscopy and egd procedures.*

Kawamoto, K., Houlihan, C. A., Balas, E. A., & Lobach, D. F. (2005). Improving clinical practice using clinical decision support systems: a systematic review of trials to identify features critical to success. *BMJ*, *330*(7494), 765.

Kazanzides, P. (2009). *Safety design for medical robots.*

Kelly, S. (2003). *Monitoring level of consciousness during anesthesia and sedation.* Newton, MA: Aspect Medical Systems, Inc.

Kenny, G. N., & Mantzaridis, H. (1999). Closed-loop control of propofol anaesthesia. *Br J Anaesth*, *83*(2), 223-8.

Kohn, L. T., Corrigan, J. M., & Donaldson, M. S. (2000). *To err is human: Building*

a safer health system. Washington, DC: The National Academies Press.

Kwoh, Y. S., Hou, J., Jonckheere, E. A., & Hayati, S. (1988). A robot with improved absolute positioning accuracy for ct guided stereotactic brain surgery. *Biomedical Engineering, IEEE Transactions on*, *35*(2), 153-160.

Lau, S., Vaknin, Z., Ramana-Kumar, A. V., Halliday, D., Franco, E. L., & Gotlieb, W. H. (2012). Outcomes and cost comparisons after introducing a robotics program for endometrial cancer surgery. *Obstet Gynecol*, *119*(4), 717-24.

Liu, N., Chazot, T., Genty, A., Landais, A., Restoux, A., McGee, K., ... Fischler, M. (2006). Titration of propofol for anesthetic induction and maintenance guided by the bispectral index: closed-loop versus manual control: a prospective, randomized, multicenter study. *Anesthesiology*, *104*(4), 686-95.

Liu, N., Chazot, T., Hamada, S., Landais, A., Boichut, N., Dussaussoy, C., ... Fischler, M. (2011). Closed-loop coadministration of propofol and remifentanil guided by bispectral index: A randomized multicenter study. *Anesthesia & Analgesia*, *112*(3), 546-557.

Liu, N., Chazot, T., Trillat, B., Michel-Cherqui, M., Marandon, J., Law-Koune, J., ... Fischler, M. (2008). Closed-loop control of consciousness during lung transplantation: an observational study. *Journal of cardiothoracic and vascular anesthesia*, *22*(4), 611–615.

Liu, N., Chazot, T., Trillat, B., Pirracchio, R., Law-Koune, J. D., Barvais, L., & Fischler, M. (2006). Feasibility of closed-loop titration of propofol guided by the bispectral index for general anaesthesia induction: a prospective randomized study. *Eur J Anaesthesiol*, *23*(6), 465-9.

Liu, N., Le Guen, M., Benabbes-Lambert, F., Chazot, T., Trillat, B., Sessler, D. I., & Fischler, M. (2012). Feasibility of closed-loop titration of propofol and remifentanil guided by the spectral m-entropy monitor. *Anesthesiology*, *116*(2), 286-95.

Liu, N., Pruszkowski, O., Leroy, J., Chazot, T., Trillat, B., Colchen, A., ... Fischler, M. (2013). Automatic administration of propofol and remifentanil guided by the bispectral index during rigid bronchoscopic procedures: a randomized trial. *Canadian Journal of Anesthesia/Journal canadien d'anesthésie*, 1-7.

Locher, S., Stadler, K., Boehlen, T., Bouillon, T., Leibundgut, D., Schumacher, P., ... Zbinden, A. (2004). A new closed-loop control system for isoflurane using bispectral index outperforms manual control. *Anesthesiology*, *101*(3), 591–602.

Lortat-Jacob, B., Billard, V., Buschke, W., & Servin, F. (2009). Assessing the clinical or pharmaco-economical benefit of target controlled desflurane delivery in surgical patients using the zeus® anaesthesia machine*. *Anaesthesia*, *64*(11), 1229-1235.

Louw, D. F., Fielding, T., McBeth, P. B., Gregoris, D., Newhook, P., & Sutherland, G. R. (2004). Surgical robotics: a review and neurosurgical prototype development. *Neurosurgery*, *54*(3), 525-537.

Luginbuhl, M., Bieniok, C., Leibundgut, D., Wymann, R., Gentilini, A., & Schnider, T. W. (2006). Closed-loop control of mean arterial blood pressure during surgery with alfentanil: clinical evaluation of a novel model-based predictive controller. *Anesthesiology*, *105*(3), 462-70.

Madhavan, J. S., Puri, G. D., & Mathew, P. J. (2011). Closed-loop isoflurane administration with bispectral index in open heart surgery: randomized controlled trial with manual control. *Acta Anaesthesiol Taiwan*, *49*(4), 130-5.

Manberg, P. J., Vozella, C. M., & Kelley, S. D. (2008). Regulatory challenges facing closed-loop anesthetic drug infusion devices. *Clinical pharmacology and therapeutics*, *84*, 166-9.

Menon, M., Tewari, A., Baize, B., Guillonneau, B., & Vallancien, G. (2002). Prospective comparison of radical retropubic prostatectomy and robot-assisted anatomic prostatectomy: the vattikuti urology institute experience. *Urology*, *60*(5), 864-8.

Miller, R. (2009). *Miller's anesthesia* (7th ed.). Churchill Livingstone/Elsevier.

Morgan, G., Mikhail, M., & Murray, M. (2006). *Clinical anesthesiology* (4th ed.). McGraw-Hill Education.

Morley, A., Derrick, J., Mainland, P., Lee, B., & Short, T. (2000). Closed loop control of anaesthesia: an assessment of the bispectral index as the target of control. *Anaesthesia*, *55*(10), 953-959.

Morse, J., Wehbe, M., Taddei, R., Cyr, S., & Hemmerling, T. M. (2013). Magellan: Technical description of a new system for robot-assisted nerve blocks. *Journal of Computers*, *8*(6), 1401-1405.

Mortier, E., Struys, M., De Smet, T., Versichelen, L., & Rolly, G. (1998). Closed-loop controlled administration of propofol using bispectral analysis. *Anaesthesia*, *53*(8), 749-54.

Nair, B. G., Newman, S., Peterson, G. N., & Schwid, H. A. (2013). Smart anesthesia manager (sam)-a real-time decision support system for anesthesia care during surgery. *Biomedical Engineering, IEEE Transactions on*, *60*(1), 207-210.

Ogata, K. (2010). *Modern control engineering* (Vol. 5). Prentice-Hall.

Olkkola, K., Schwilden, H., & Apffelstaedt, C. (1991). Modelbased adaptive closed-loop feedback control of atracuriuminduced neuromuscular blockade. *Acta anaesthesiologica scandinavica*, *35*(5), 420-423.

Pambianco, D. J., Vargo, J. J., Pruitt, R. E., Hardi, R., & Martin, J. F. (2011). Computer-assisted personalized sedation for upper endoscopy and colonoscopy: a comparative, multicenter randomized study. *Gastrointest Endosc*, *73*(4), 765-72.

Pambianco, D. J., Whitten, C. J., Moerman, A., Struys, M. M., & Martin, J. F. (2008). An assessment of computer-assisted personalized sedation: a sedation delivery system to administer propofol for gastrointestinal endoscopy. *Gastrointest Endosc*, *68*(3), 542-7.

Patek, S., Bequette, B., Breton, M., Buckingham, B., Dassau, E., Doyle, F., ... Zisser, H. (2009). In silico preclinical trials: methodology and engineering guide to closed-loop control in type 1 diabetes mellitus. *Journal of diabetes science and technology*, *3*(2), 269–282.

Philips. (2011). Data export interface programming guide (Rev. H.0 ed.) [Computer software manual].

Puri, G., Kumar, B., & Aveek, J. (2007). Closed-loop anaesthesia delivery system (clads) using bispectral index: a performance assessment study. *Anaesthesia & Intensive Care*, *35*(3), 357 - 362.

Puri, G. D., Jayant, A., Tsering, M., Dorje, M., & Tashi, M. (2012). Closed loop anaesthesia at high altitude (3505 m above sea level): Performance characteristics of an indigenously developed closed loop anaesthesia delivery system. *Indian J Anaesth*, *56*(3), 238-42.

Ramsay, C., Pickard, R., Robertson, C., Close, A., Vale, L., Armstrong, N., ... Soomro, N. (2012). Systematic review and economic modelling of the relative clinical benefit and cost-effectiveness of laparoscopic surgery and robotic surgery for removal of the prostate in men with localised prostate cancer. *Health Technol Assess*, *16*(41), 1-313.

Rashid, T. G., Kini, M., & Ind, T. E. (2010). Comparing the learning curve for robotically assisted and straight stick laparoscopic procedures in surgical novices. *Int J Med Robot*, *6*(3), 306-10.

Reason, J. (2000). Human error: Models and management. *BMJ: British Medical Journal*, *320*(7237), 768-770.

Rinehart, J., Liu, N., Alexander, B., & Cannesson, M. (2012). Closed-loop systems in anesthesia: Is there a potential for closed-loop fluid management and hemodynamic optimization? *Anesthesia & Analgesia*, *114*, 130-143.

Robinson, D. H., & Toledo, A. H. (2012). Historical development of modern anesthesia. *Journal of Investigative Surgery*, *25*(3), 141–149.

Satava, R. M. (2002). Surgical robotics: the early chronicles: a personal historical perspective. *Surg Laparosc Endosc Percutan Tech*, *12*(1), 6-16.

Schiffman, G. (2014). *Endotracheal intubation.* Retrieved February 2014, from `http://www.medicinenet.com/endotracheal_intubation/article.htm`

Schwilden, H. (1981). A general method for calculating the dosage scheme in linear pharmacokinetics. *European journal of clinical pharmacology*, *20*(5), 379-386.

Schwilden, H., Schuttler, J., & Stoeckel, H. (1987). Closed-loop feedback control of methohexital anesthesia by quantitative eeg analysis in humans. *Anesthesiology*,

67(3), 341-7.

Schwilden, H., Stoeckel, H., & Schuttler, J. (1989). Closed-loop feedback control of propofol anaesthesia by quantitative eeg analysis in humans. *Br J Anaesth*, *62*(3), 290-6.

Sieber, T. J., Frei, C. W., Derighetti, M., Feigenwinter, P., Leibundgut, D., & Zbinden, A. M. (2000). Model-based automatic feedback control versus human control of end-tidal isoflurane concentration using low-flow anaesthesia. *Br J Anaesth*, *85*(6), 818-25.

Simanski, O., Janda, M., Schubert, A., Bajorat, J., Hofmockel, R., & Lampe, B. (2009). Progress of automatic drug delivery in anaesthesia—the 'rostock assistant system for anaesthesia control (ran)'. *International Journal of Adaptive Control and Signal Processing*, *23*(5), 504-521.

Solanki, A., Puri, G. D., & Mathew, P. J. (2010). Bispectral index-controlled postoperative sedation in cardiac surgery patients: a comparative trial between closed loop and manual administration of propofol. *Eur J Anaesthesiol*, *27*(8), 708-13.

Strode, S. W., Gustke, S., & Allen, A. (1999). Technical and clinical progress in telemedicine. *JAMA*, *281*(12), 1066-1068.

Struys, M., De Smet, T., Versichelen, L., Van De Velde, S., Van den Broecke, R., & Mortier, E. (2001). Comparison of closed-loop controlled administration of propofol using bispectral index as the controlled variable versus "standard practice" controlled administration. *Anesthesiology*, *95*(1), 6-17.

Struys, M. M., De Smet, T., Versichelen, L. F., Van De Velde, S., Van den Broecke,

R., & Mortier, E. P. (2001). Comparison of closed-loop controlled administration of propofol using bispectral index as the controlled variable versus "standard practice" controlled administration. *Anesthesiology*, *95*(1), 6-17.

Tighe, P., Badiyan, S., Luria, I., Boezaart, A., & Parekattil, S. (2010). Robot-assisted regional anesthesia: A simulated demonstration. *Anesthesia & Analgesia*, *111*(3), 813-816.

Tighe, P., Badiyan, S., Luria, I., Lampotang, S., & Parekattil, S. (2010). Robot-assisted airway support: A simulated case. *Anesthesia & Analgesia*, *111*(4), 929-931.

Tomaszewski, J. J., Matchett, J. C., Davies, B. J., Jackman, S. V., Hrebinko, R. L., & Nelson, J. B. (2012). Comparative hospital cost-analysis of open and robotic-assisted radical prostatectomy. *Urology*, *80*(1), 126-9.

Varvel, J., Donoho, D., & Shafer, S. (1992). Measuring the predictive performance of computer-controlled infusion pumps. *Journal of Pharmacokinetics and Pharmacodynamics*, *20*(1), 63–94.

Viby-Mogensen, J. (1982). Clinical assessment of neuromuscular transmission. *British Journal of Anaesthesia*, *54*(2), 209-223.

VMware, Inc. (2013). *Vmware® vcloud® architecture toolkit cloud bursting* (Tech. Rep.). VMware, Inc.

Wang, H. E., Kupas, D. F., Paris, P. M., Bates, R. R., Costantino, J. P., & Yealy, D. M. (2003). Multivariate predictors of failed prehospital endotracheal intubation. *Academic emergency medicine*, *10*(7), 717-724.

Wang, H. E., Sweeney, T. A., O'Connor, R. E., & Rubinstein, H. (2001). Failed prehospital intubations: an analysis of emergency department courses and outcomes. *Prehosp Emerg Care*, *5*(2), 134-41.

Wehbe, M., Arbeid, E., Cyr, S., Mathieu, P., Taddei, R., Morse, J., & Hemmerling, T. (2013). A technical description of a novel pharmacological anesthesia robot. *Journal of Clinical Monitoring and Computing*, 1-8.

Wehbe, M., Mathieu, P. A., & Hemmerling, T. M. (2012). Relaxofon: a neuromuscular blockade monitor for patients under general anesthesia. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society.*.

Westenskow, D. R., Orr, J. A., Simon, F. H., Bender, H.-J., & Frankenberger, H. (1992). Intelligent alarms reduce anesthesiologist's response time to critical faults. *Anesthesiology*, *77*(6), 1074-1079.

Willis, D. L., Gonzalgo, M. L., Brotzman, M., Feng, Z., Trock, B., & Su, L. M. (2012). Comparison of outcomes between pure laparoscopic vs robot-assisted laparoscopic radical prostatectomy: a study of comparative effectiveness based upon validated quality of life outcomes. *BJU Int*, *109*(6), 898-905.

Wong, D. T., Kamming, D., Salenieks, M. E., Go, K., Kohm, C., & Chung, F. (2004). Preadmission anesthesia consultation using telemedicine technology: A pilot study. *Anesthesiology*, *100*(6), 1605-1607.

Yohannes, P., Rotariu, P., Pinto, P., Smith, A. D., & Lee, B. R. (2002). Comparison of robotic versus laparoscopic skills: is there a difference in the learning curve? *Urology*, *60*(1), 39–45.

Zaouter, C., Wehbe, M., Cyr, S., Morse, J., Taddei, R., Mathieu, P., & Hemmerling, T. (2013). Use of a decision support system improves the management of hemodynamic and respiratory events in orthopedic patients under propofol sedation and spinal analgesia: a randomized trial. *Journal of Clinical Monitoring and Computing*, 1-7.

Appendices

APPENDIX A
Intellivue Data Export Interface communication protocol: C# source code

A.1 IntellivueBridge.cs

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Net;
6  using System.Net.Sockets;
7  using System.ComponentModel;
8
9  namespace IntellivueBridge
10 {
11     /// <summary>
12     /// Handles connections with a Philips Intellivue Monitor.
13     /// </summary>
14     public class IntellivueBridge
15     {
16         /* Notes on connection process */
17         /* Step 1: Assign an IP address to the monitor (handled by BootPServer project).
18          * Step 2: Receive a Connection Indication Event from the monitor.
19          * Step 3: Send an Association Request to the monitor.
20          * Step 4: Receive an Association Response from the monitor.
21          * Step 5: Receive a MDS Create Event packet from the monitor.
22          * Step 6: Send a MDS Create Event Result packet to the monitor.
23          * Step 7: Send Poll Requests and receive Poll Results as necessary.
24          * Step 8: Send Release Request and receive Release Response to release connection with monitor.
25          */
26
27         /* Members */
28         private BackgroundWorker _connBGW;          // Handles sending and receiving of packets with Intellivue
29                                                     // monitor for receiving the Connection Indication Event
30         private int _listenPort;                    // Defines the port to use for all messages after Connection Indica
31         private BackgroundWorker _networkBGW;       // Handles sending and receiving of packets with Intellivue monitor
32                                                     // Indication Event is received
33         private UdpClient _uc;                      // Handles all sending and receiving of packets with
34                                                     // Intellivue monitor
35         private System.Timers.Timer _timer;         // Timer for requesting data at fixed intervals
36         private IntellivueBridgeStatuses _status = IntellivueBridgeStatuses.Disconnected;   // Status
```

```
37          BootPServer.BootPServer _bootPServer;       // The BootP Server used to issue DHCP reservation
38                                                      // to Intellivue monitor
39
40          private short _pollNumber = 1;
41          private int _pollInterval = 1000;          // Interval at which we poll for data
42
43          /* Delegates */
44          public delegate void AirwayMacConcentrationETReceivedHandler(object sender,
45                                                                       FloatValueReceivedEventArgs args);
46          public delegate void AirwayMacConcentrationInspReceivedHandler(object sender,
47                                                                       FloatValueReceivedEventArgs args);
48          public delegate void ArterialBloodPressureReceivedHandler(object sender,
49                                                                       BloodPressureReceivedEventArgs args);
50          public delegate void BisReceivedHandler(object sender, BisReceivedEventArgs args);
51          public delegate void CO2ETReceivedHandler(object sender, FloatValueReceivedEventArgs args);
52          public delegate void EmgReceivedHandler(object sender, EmgReceivedEventArgs args);
53          public delegate void FiO2ReceivedHandler(object sender, IntegerValueReceivedEventArgs args);
54          public delegate void HeartRateReceivedHandler(object sender, HeartRateReceivedEventArgs args);
55          public delegate void ieDESReceivedHandler(object sender, IntegerValueReceivedEventArgs args);
56          public delegate void IERatioReceivedHandler(object sender, IntegerValueReceivedEventArgs args);
57          public delegate void ieSEVReceivedHandler(object sender, IntegerValueReceivedEventArgs args);
58          public delegate void MnAwPReceivedHandler(object sender, IntegerValueReceivedEventArgs args);
59          public delegate void HematocritReceivedHandler(object sender, IntegerValueReceivedEventArgs args);
60          public delegate void HemoglobinReceivedHandler(object sender, IntegerValueReceivedEventArgs args);
61          public delegate void NonInvasiveBloodPressureReceivedHandler(object sender,
62                                                                       BloodPressureReceivedEventArgs args);
63          public delegate void OutputTextChangedHandler(object sender, OutputTextChangedEventArgs args);
64          public delegate void OxygenConsumptionVO2ReceivedHandler(object sender, IntegerValueReceivedEventArgs args);
65          public delegate void PerfusionIndicatorReceivedHandler(object sender, FloatValueReceivedEventArgs args);
66          public delegate void PlateauPressureReceivedHandler(object sender, IntegerValueReceivedEventArgs args);
67          public delegate void PositiveEndExpiratoryPressureReceivedHandler(object sender,
68                                                                       IntegerValueReceivedEventArgs args);
69          public delegate void PositiveInspiratoryPressureReceivedHandler(object sender,
70                                                                       IntegerValueReceivedEventArgs args);
71          public delegate void PulseFromNbpReceivedHandler(object sender, PulseReceivedEventArgs args);
72          public delegate void PulsePlethReceivedHandler(object sender, PulseReceivedEventArgs args);
73          public delegate void RespiratoryRateReceivedHandler(object sender, RespiratoryRateReceivedEventArgs args);
74          public delegate void SpO2ReceivedHandler(object sender, SpO2ReceivedEventArgs args);
75          public delegate void SqiReceivedHandler(object sender, SqiReceivedEventArgs args);
76          public delegate void SuppressionRatioReceivedHandler(object sender, IntegerValueReceivedEventArgs args);
77          public delegate void StatusChangedHandler(object sender, StatusChangedEventArgs args);
78          public delegate void TemperatureUnspecifiedReceivedHandler(object sender, TemperatureReceivedEventArgs args);
79          public delegate void TidalVolumeReceivedHandler(object sender, IntegerValueReceivedEventArgs args);
80          public delegate void TidalVolumeInspiredReceivedHandler(object sender, IntegerValueReceivedEventArgs args);
81
82          /* Events for notifying of received data */
83          /// <summary>
```

```
84          /// Occurs when an Airway MAC Concentration ET value is received from the Intellivue monitor.
85          /// </summary>
86          public event AirwayMacConcentrationETReceivedHandler AirwayMacConcentrationEtReceived = delegate { };
87
88          /// <summary>
89          /// Occurs when an Airway MAC Concentration Insp value is received from the Intellivue monitor.
90          /// </summary>
91          public event AirwayMacConcentrationInspReceivedHandler AirwayMacConcentrationInspReceived = delegate { };
92
93          /// <summary>
94          /// Occurs when an Arterial Blood Pressure value is received from the Intellivue monitor.
95          /// </summary>
96          public event ArterialBloodPressureReceivedHandler ArterialBloodPressureReceived = delegate { };
97
98          /// <summary>
99          /// Occurs when a BIS value is received from the Intellivue monitor.
100         /// </summary>
101         public event BisReceivedHandler BisReceived = delegate { };
102
103         /// <summary>
104         /// Occurs when a CO2 value is received from the Intellivue monitor.
105         /// </summary>
106         public event CO2ETReceivedHandler CO2ETReceived = delegate { };
107
108         /// <summary>
109         /// Occurs when a EMG value is received from the Intellivue monitor.
110         /// </summary>
111         public event EmgReceivedHandler EmgReceived = delegate { };
112
113         /// <summary>
114         /// Occurs when a FiO2 value is received from the intellivue monitor
115         /// </summary>
116         public event FiO2ReceivedHandler FiO2Received = delegate { };
117
118         /// <summary>
119         /// Occurs when a ieDES value is received from the Intellivue monitor.
120         /// </summary>
121         public event ieDESReceivedHandler ieDESReceived = delegate { };
122
123         /// <summary>
124         /// Occurs when a IERatio value is received from the Intellivue monitor.
125         /// </summary>
126         public event IERatioReceivedHandler IERatioReceived = delegate { };
127
128         /// <summary>
129         /// Occurs when a ieSEV value is received from the Intellivue monitor.
130         /// </summary>
```

```
131         public event ieSEVReceivedHandler ieSEVReceived = delegate { };
132
133         /// <summary>
134         /// Occurs when a MnAwP value is received from the intellivue monitor
135         /// </summary>
136         public event MnAwPReceivedHandler MnAwPReceived = delegate { };
137
138         /// <summary>
139         /// Occurs when a Heart Rate value is received from the Intellivue monitor.
140         /// </summary>
141         public event HeartRateReceivedHandler HeartRateReceived = delegate { };
142
143         /// <summary>
144         /// Occurs when a Hematocrit value is received from the Intellivue monitor.
145         /// </summary>
146         public event HematocritReceivedHandler HematocritReceived = delegate { };
147
148         /// <summary>
149         /// Occurs when a Hemoglobin value is received from the Intellivue monitor.
150         /// </summary>
151         public event HemoglobinReceivedHandler HemoglobinReceived = delegate { };
152
153         /// <summary>
154         /// Occurs when a Non-Invasive Blood Pressure value is received from the Intellivue monitor.
155         /// </summary>
156         public event NonInvasiveBloodPressureReceivedHandler NonInvasiveBloodPressureReceived = delegate { };
157
158         /// <summary>
159         /// Occurs when output is output.
160         /// </summary>
161         public event OutputTextChangedHandler OutputTextChanged = delegate { };
162
163         /// <summary>
164         /// Occurs when an Oxygen Consumption VO2 value is received from the Intellivue monitor.
165         /// </summary>
166         public event OxygenConsumptionVO2ReceivedHandler OxygenConsumptionVO2Received = delegate { };
167
168         /// <summary>
169         /// Occurs when a Perfusion Indicator value is received from the Intellivue monitor.
170         /// </summary>
171         public event PerfusionIndicatorReceivedHandler PerfusionIndicatorReceived = delegate { };
172
173         /// <summary>
174         /// Occurs when a Plateau Pressure value is received from the Intellivue monitor.
175         /// </summary>
176         public event PlateauPressureReceivedHandler PlateauPressureReceived = delegate { };
177
```

```
178         /// <summary>
179         /// Occurs when a Positive End Expiratory Pressure value is received from the Intellivue monitor.
180         /// </summary>
181         public event PositiveEndExpiratoryPressureReceivedHandler
182                                                             PositiveEndExpiratoryPressureReceived = delegate { };
183
184         /// <summary>
185         /// Occurs when a Positive Inspiratory Pressure value is received from the Intellivue monitor.
186         /// </summary>
187         public event PositiveInspiratoryPressureReceivedHandler PositiveInspiratoryPressureReceived = delegate { };
188
189         /// <summary>
190         /// Occurs when a Pulse is received from the Non-Invasive Blood Pressure.
191         /// </summary>
192         public event PulseFromNbpReceivedHandler PulseFromNonInvasiveBloodPressureReceived = delegate { };
193
194         /// <summary>
195         /// Occurs when a Pulse is received from the Plethysmograph.
196         /// </summary>
197         public event PulsePlethReceivedHandler PulsePlethReceived = delegate { };
198
199         /// <summary>
200         /// Occurs when a Respiratory Rate value is received from the Intellivue monitor.
201         /// </summary>
202         public event RespiratoryRateReceivedHandler RespiratoryRateReceived = delegate { };
203
204         /// <summary>
205         /// Occurs when a SpO2 value is received from the Intellivue monitor.
206         /// </summary>
207         public event SpO2ReceivedHandler SpO2Received = delegate { };
208
209         /// <summary>
210         /// Occurs when a SQI value is received from the Intellivue monitor.
211         /// </summary>
212         public event SqiReceivedHandler SqiReceived = delegate { };
213
214         /// <summary>
215         /// Occurs when a status changes.
216         /// </summary>
217         public event StatusChangedHandler StatusChanged = delegate { };
218
219         /// <summary>
220         /// Occurs when a Suppression Ratio value is received from the Intellivue monitor.
221         /// </summary>
222         public event SuppressionRatioReceivedHandler SuppressionRatioReceived = delegate { };
223
224         /// <summary>
```

```
225         /// Occurs when an Unspecified Temperature value is received from the Intellivue monitor.
226         /// </summary>
227         public event TemperatureUnspecifiedReceivedHandler TemperatureUnspecifiedReceived = delegate { };
228
229         /// <summary>
230         /// Occurs when a Tidal Volume value is received from the Intellivue monitor.
231         /// </summary>
232         public event TidalVolumeReceivedHandler TidalVolumeReceived = delegate { };
233
234         /// <summary>
235         /// Occurs when a Tidal Volume Inspired value is received from the Intellivue monitor.
236         /// </summary>
237         public event TidalVolumeInspiredReceivedHandler TidalVolumeInspiredReceived = delegate { };
238
239         /* Properties */
240         public float AirwayMacConcentrationET { get; private set; }
241         public float AirwayMacConcentrationInsp { get; private set; }
242         public int ArterialDiastolic { get; private set; }
243         public int ArterialSystolic { get; private set; }
244         public int ArterialMAP { get; private set; }
245         public int BIS { get; private set; }
246         public float CO2ET { get; private set; }
247         public int EMG { get; private set; }
248         public int FiO2 { get; private set; }
249         public int HeartRate { get; private set; }
250         public int ieDES { get; private set; }
251         public int IERatio { get; private set; }
252         public int ieSEV { get; private set; }
253         public int MnAwP { get; private set; }
254         public int Hematocrit { get; private set; }
255         public int Hemoglobin { get; private set; }
256         public int NonInvasiveDiastolic { get; private set; }
257         public int NonInvasiveSystolic { get; private set; }
258         public int NonInvasiveMAP { get; private set; }
259         public int OxygenConsumptionVO2 { get; private set; }
260         public float PerfusionIndicator { get; private set; }
261         public int PlateauPressure { get; private set; }
262         public int PositiveEndExpiratoryPressure { get; private set; }
263         public int PositiveInspiratoryPressure { get; private set; }
264         public int PulseFromNBP { get; private set; }
265         public int PulsePleth { get; private set; }
266         public float RespiratoryRate { get; private set; }
267         public float SpO2 { get; private set; }
268         public int SQI { get; private set; }
269         public int SuppressionRatio { get; private set; }
270         public float TemperatureUnspecified { get; private set; }
271         public Units TemperatureUnspecifiedUnits { get; private set; }
```

```
272         public int TidalVolume { get; private set; }
273         public int TidalVolumeInspired { get; private set; }
274         public string OutputText { get; private set; }
275 
276         /// <summary>
277         /// Gets the interval at which requests are made to the Intellivue monitor for data.
278         /// This interval is in milliseconds and defaults to 1000.
279         /// </summary>
280         public int PollInterval
281         {
282             get
283             {
284                 return _pollInterval;
285             }
286         }
287 
288         /// <summary>
289         /// Gets the status of the Intellivue Bridge.
290         /// </summary>
291         public IntellivueBridgeStatuses Status
292         {
293             get
294             {
295                 return _status;
296             }
297             private set
298             {
299                 if (_status != value)
300                 {
301                     StatusChanged(this, new StatusChangedEventArgs(value));
302                     Output("Status changed from " + _status + " to: " + value);
303                     _status = value;
304                 }
305             }
306         }
307 
308         /* Constructors */
309         /// <summary>
310         /// Default constructor.  Make sure Constants.InterfaceIP is assigned to an enabled adapter
311         /// that is connected to the network before calling and that ConnectIndicationEvent.ListenPort is available.
312         /// </summary>
313         public IntellivueBridge()
314         {
315             // Create UDP client for receiving connect indication event
316             try
317             {
318                 _uc = new UdpClient(new IPEndPoint(IPAddress.Parse(Constants.InterfaceIP),
```

```
319                                             ConnectIndicationEvent.ListenPort));
320                 _uc.Client.ReceiveTimeout = Constants.ConnectionIndicationEventTimeout;
321                 _uc.Client.SendTimeout = Constants.ConnectionIndicationEventTimeout;
322             }
323             catch (SocketException soe)
324             {
325                 throw soe;
326             }
327 
328             // Configure and run background worker for receiving connect indication event
329             _connBGW = new BackgroundWorker() { WorkerReportsProgress = true, WorkerSupportsCancellation = true };
330             _connBGW.DoWork += connBGW_DoWork;
331             _connBGW.RunWorkerCompleted += new RunWorkerCompletedEventHandler(connBGW_RunWorkerCompleted);
332             _connBGW.ProgressChanged += new ProgressChangedEventHandler(connBGW_ProgressChanged);
333 
334             // Configure background worker for handling network communication
335             _networkBGW = new BackgroundWorker() { WorkerReportsProgress = true, WorkerSupportsCancellation = true };
336             _networkBGW.DoWork += nbw_DoWork;
337             _networkBGW.ProgressChanged += new ProgressChangedEventHandler(nbw_ProgressChanged);
338 
339             // Configure BootP Server
340             _bootPServer = new BootPServer.BootPServer();
341             _bootPServer.StatusChanged +=
342                             new BootPServer.BootPServer.StatusChangedHandler(bootPServer_StatusChanged);
343             _bootPServer.OutputTextChanged +=
344                             new BootPServer.BootPServer.OutputTextChangedHandler(bootPServer_OutputTextChanged);
345             _bootPServer.BroadcastIP = Constants.BroadcastIP;
346             _bootPServer.InterfaceIP = Constants.InterfaceIP;
347             _bootPServer.TargetIP = Constants.IntellivueMonitorHostName;
348 
349             // Create timer for requesting data at intervals
350             _timer = new System.Timers.Timer(_pollInterval);
351             _timer.Elapsed += new System.Timers.ElapsedEventHandler(timer_Elapsed);
352 
353             // Set default status
354             Status = IntellivueBridgeStatuses.Disconnected;
355 
356             // Set default statuses
357             SetDefaultProperties();
358         }
359 
360         /* Methods */
361         void connBGW_RunWorkerCompleted(object sender, RunWorkerCompletedEventArgs e)
362         {
363             // Close UDP client used for receiving connect indication event
364             _uc.Close();
365 
```

```
366             if (_listenPort > 0)
367             {
368                 // Create new UDP client for receiving all other packets
369                 _uc = new UdpClient(new IPEndPoint(IPAddress.Parse(Constants.InterfaceIP), _listenPort));
370                 _uc.Client.ReceiveTimeout = Constants.DataPollTimeout;
371                 _uc.Client.SendTimeout = Constants.DataPollTimeout;
372 
373                 // Connect to monitor
374                 //_uc.Connect(IPAddress.Parse(Constants.IntellivueMonitorHostName), _listenPort);
375 
376                 // Connection complete
377                 Status = IntellivueBridgeStatuses.Connected;
378 
379                 // Start background monitor for handling network communication with monitor
380                 _networkBGW.RunWorkerAsync();
381 
382                 // Give time to other thread to get ready to receive Association Response
383                 System.Threading.Thread.Sleep(500);
384 
385                 // Send Association Request Message (_networkBGW will handle Association Response)
386                 AssociationRequest ar = new AssociationRequest();
387                 _uc.Send(ar.ToByteArray(), ar.Length,
388                             new IPEndPoint(IPAddress.Parse(Constants.IntellivueMonitorHostName), _listenPort));
389                 Output("Sending Assocation Request.");
390             }
391             else
392             {
393                 Output("Invalid local port: did a timeout occur?");
394                 Shutdown(false);
395             }
396         }
397 
398         /// <summary>
399         /// Handles connecting with the Intellivue Monitor
400         /// </summary>
401         private void connBGW_DoWork(object sender, DoWorkEventArgs e)
402         {
403             while (!_connBGW.CancellationPending)
404             {
405                 try
406                 {
407                     byte[] packet = new byte[Constants.MaximumPacketLength];
408 
409                     // Receive a packet from the Intellivue monitor
410                     _uc.EnableBroadcast = true;
411                     IPEndPoint externalIpEndPoint = new IPEndPoint(IPAddress.Any,
412                         ConnectIndicationEvent.ListenPort);
```

```
413                     packet = _uc.Receive(ref externalIpEndPoint);
414
415                     if (GetPacketType(packet) == PacketTypes.ConnectionIndicationEvent)
416                     {
417                         ConnectIndicationEvent cie = new ConnectIndicationEvent(packet);
418
419                         if (cie.DataExportProtocolPort > 0)
420                         {
421                             // The port was successfully parsed, so we are done receiving Connection Indication Event
422                             _listenPort = cie.DataExportProtocolPort;
423                             _connBGW.ReportProgress(100,
424                               String.Format("Received Connection Indication Event. Data Export Protocol Port: {0}",
425                               _listenPort));
426                             e.Result = true;
427                             return;
428                         }
429                     }
430
431                     Output("Received a packet from the monitor but it isn't a Connection Indication Event.");
432                 }
433                 catch (SocketException soe)
434                 {
435                     if (soe.SocketErrorCode == SocketError.TimedOut)
436                     {
437                         // Our connection timed out
438                         Output("Error: Socket connection timed out while waiting for Connection Indication Event.");
439                         Shutdown(false);
440                         return;
441                     }
442                     else
443                     {
444                         Output("Error: Socket Exception occurred while waiting for Connection Indication Event.");
445                         Shutdown(false);
446                         return;
447                     }
448                 }
449             }
450         }
451
452         /// <summary>
453         /// Handles reporting changes in connection progress.
454         /// </summary>
455         private void connBGW_ProgressChanged(object sender, ProgressChangedEventArgs e)
456         {
457             Output(e.UserState.ToString());
458         }
459
```

```
460         /// <summary>
461         /// Handles the reception of a Poll Data Response.
462         /// </summary>
463         private void HandlePollDataResponse(PollDataResponse pdr)
464         {
465             if (pdr != null)
466             {
467                 if (pdr.AirwayMacConcentrationET != Constants.FloatValueUnset)
468                 {
469                     AirwayMacConcentrationEtReceived(this,
470                             new FloatValueReceivedEventArgs(pdr.AirwayMacConcentrationET));
471                     this.AirwayMacConcentrationET = pdr.AirwayMacConcentrationET;
472                 }
473                 if (pdr.AirwayMacConcentrationInsp != Constants.FloatValueUnset)
474                 {
475                     AirwayMacConcentrationInspReceived(this,
476                             new FloatValueReceivedEventArgs(pdr.AirwayMacConcentrationInsp));
477                     this.AirwayMacConcentrationInsp = pdr.AirwayMacConcentrationInsp;
478                 }
479                 if (pdr.ArterialDiastolic != Constants.IntValueUnset
480                             && pdr.ArterialMAP != Constants.IntValueUnset
481                             && pdr.ArterialSystolic != Constants.IntValueUnset)
482                 {
483                     ArterialBloodPressureReceived(this,
484                             new BloodPressureReceivedEventArgs(pdr.ArterialDiastolic,
485                                     pdr.ArterialSystolic, pdr.ArterialMAP));
486                     this.ArterialDiastolic = pdr.ArterialDiastolic;
487                     this.ArterialMAP = pdr.ArterialMAP;
488                     this.ArterialSystolic = pdr.ArterialSystolic;
489                 }
490                 if (pdr.BIS != Constants.IntValueUnset)
491                 {
492                     BisReceived(this, new BisReceivedEventArgs(pdr.BIS));
493                     this.BIS = pdr.BIS;
494                 }
495                 if (pdr.CO2ET != Constants.FloatValueUnset)
496                 {
497                     CO2ETReceived(this, new FloatValueReceivedEventArgs(pdr.CO2ET));
498                     this.CO2ET = pdr.CO2ET;
499                 }
500                 if (pdr.EMG != Constants.IntValueUnset)
501                 {
502                     EmgReceived(this, new EmgReceivedEventArgs(pdr.EMG));
503                     this.EMG = pdr.EMG;
504                 }
505                 if (pdr.SQI != Constants.IntValueUnset)
506                 {
```

```
507                     SqiReceived(this, new SqiReceivedEventArgs(pdr.SQI));
508                     this.SQI = pdr.SQI;
509                 }
510                 if (pdr.FiO2 != Constants.IntValueUnset)
511                 {
512                     FiO2Received(this, new IntegerValueReceivedEventArgs(pdr.FiO2));
513                     this.FiO2 = pdr.FiO2;
514                 }
515                 if (pdr.ieDES != Constants.IntValueUnset)
516                 {
517                     ieDESReceived(this, new IntegerValueReceivedEventArgs(pdr.ieDES));
518                     this.ieDES = pdr.ieDES;
519                 }
520                 if (pdr.IERation != Constants.IntValueUnset)
521                 {
522                     IERatioReceived(this, new IntegerValueReceivedEventArgs(pdr.IERation));
523                     this.IERatio = pdr.IERation;
524                 }
525                 if (pdr.ieSEV != Constants.IntValueUnset)
526                 {
527                     ieSEVReceived(this, new IntegerValueReceivedEventArgs(pdr.ieSEV));
528                     this.ieSEV = pdr.ieSEV;
529                 }
530                 if (pdr.MnAwP != Constants.IntValueUnset)
531                 {
532                     MnAwPReceived(this, new IntegerValueReceivedEventArgs(pdr.MnAwP));
533                     this.MnAwP = pdr.MnAwP;
534                 }
535                 if (pdr.NonInvasiveDiastolic != Constants.IntValueUnset
536                                 && pdr.NonInvasiveMAP != Constants.IntValueUnset
537                                 && pdr.NonInvasiveSystolic != Constants.IntValueUnset)
538                 {
539                     NonInvasiveBloodPressureReceived(this,
540                                 new BloodPressureReceivedEventArgs(pdr.NonInvasiveDiastolic,
541                                         pdr.NonInvasiveSystolic, pdr.NonInvasiveMAP));
542                     this.NonInvasiveDiastolic = pdr.NonInvasiveDiastolic;
543                     this.NonInvasiveMAP = pdr.NonInvasiveMAP;
544                     this.NonInvasiveSystolic = pdr.NonInvasiveSystolic;
545                 }
546                 if (pdr.HeartRate != Constants.IntValueUnset)
547                 {
548                     HeartRateReceived(this, new HeartRateReceivedEventArgs(pdr.HeartRate));
549                     this.HeartRate = pdr.HeartRate;
550                 }
551                 if (pdr.Hematocrit != Constants.IntValueUnset)
552                 {
553                     HematocritReceived(this, new IntegerValueReceivedEventArgs(pdr.Hematocrit));
```

```
554                     this.Hematocrit = pdr.Hematocrit;
555                 }
556                 if (pdr.Hemoglobin != Constants.IntValueUnset)
557                 {
558                     HemoglobinReceived(this, new IntegerValueReceivedEventArgs(pdr.Hemoglobin));
559                     this.Hemoglobin = pdr.Hemoglobin;
560                 }
561                 if (pdr.OxygenConsumptionVO2 != Constants.IntValueUnset)
562                 {
563                     OxygenConsumptionVO2Received(this, new IntegerValueReceivedEventArgs(pdr.OxygenConsumptionVO2));
564                     this.OxygenConsumptionVO2 = pdr.OxygenConsumptionVO2;
565                 }
566                 if (pdr.PerfusionIndicator != Constants.FloatValueUnset)
567                 {
568                     PerfusionIndicatorReceived(this, new FloatValueReceivedEventArgs(pdr.PerfusionIndicator));
569                     this.PerfusionIndicator = pdr.PerfusionIndicator;
570                 }
571                 if (pdr.PlateauPressure != Constants.IntValueUnset)
572                 {
573                     PlateauPressureReceived(this, new IntegerValueReceivedEventArgs(pdr.PlateauPressure));
574                     this.PlateauPressure = pdr.PlateauPressure;
575                 }
576                 if (pdr.PositiveEndExpiratoryPressure != Constants.IntValueUnset)
577                 {
578                     PositiveEndExpiratoryPressureReceived(this,
579                                         new IntegerValueReceivedEventArgs(pdr.PositiveEndExpiratoryPressure));
580                     this.PositiveEndExpiratoryPressure = pdr.PositiveEndExpiratoryPressure;
581                 }
582                 if (pdr.PositiveInspiratoryPressure != Constants.IntValueUnset)
583                 {
584                     PositiveInspiratoryPressureReceived(this,
585                                         new IntegerValueReceivedEventArgs(pdr.PositiveInspiratoryPressure));
586                     this.PositiveInspiratoryPressure = pdr.PositiveInspiratoryPressure;
587                 }
588                 if (pdr.PulseFromNBP != Constants.IntValueUnset)
589                 {
590                     PulseFromNonInvasiveBloodPressureReceived(this, new PulseReceivedEventArgs(pdr.PulseFromNBP));
591                     this.PulseFromNBP = pdr.PulseFromNBP;
592                 }
593                 if (pdr.PulsePleth != Constants.IntValueUnset)
594                 {
595                     PulsePlethReceived(this, new PulseReceivedEventArgs(pdr.PulsePleth));
596                     this.PulsePleth = pdr.PulsePleth;
597                 }
598                 if (pdr.RespiratoryRate != Constants.FloatValueUnset)
599                 {
600                     RespiratoryRateReceived(this, new RespiratoryRateReceivedEventArgs(pdr.RespiratoryRate));
```

```
601                     this.RespiratoryRate = pdr.RespiratoryRate;
602                 }
603                 if (pdr.SpO2 != Constants.FloatValueUnset)
604                 {
605                     SpO2Received(this, new SpO2ReceivedEventArgs(pdr.SpO2));
606                     this.SpO2 = pdr.SpO2;
607                 }
608                 if (pdr.SuppressionRatio != Constants.IntValueUnset)
609                 {
610                     SuppressionRatioReceived(this, new IntegerValueReceivedEventArgs(pdr.SuppressionRatio));
611                     this.SuppressionRatio = pdr.SuppressionRatio;
612                 }
613                 if (pdr.TemperatureUnspecified != Constants.IntValueUnset)
614                 {
615                     TemperatureUnspecifiedReceived(this, new TemperatureReceivedEventArgs(pdr.TemperatureUnspecified, p
616                     this.TemperatureUnspecified = pdr.TemperatureUnspecified;
617                     this.TemperatureUnspecifiedUnits = pdr.TemperatureUnspecifiedUnits;
618                 }
619                 if (pdr.TidalVolume != Constants.IntValueUnset)
620                 {
621                     TidalVolumeReceived(this, new IntegerValueReceivedEventArgs(pdr.TidalVolume));
622                     this.TidalVolume = pdr.TidalVolume;
623                 }
624                 if (pdr.TidalVolumeInspired != Constants.IntValueUnset)
625                 {
626                     TidalVolumeInspiredReceived(this, new IntegerValueReceivedEventArgs(pdr.TidalVolumeInspired));
627                     this.TidalVolumeInspired = pdr.TidalVolumeInspired;
628                 }
629             }
630         }
631 
632         /// <summary>
633         /// Returns the packet type of (packet).
634         /// </summary>
635         internal static PacketTypes GetPacketType(byte[] packet)
636         {
637             // Determine packet type
638             if (PollDataResponse.IsPollDataResponse(packet))
639             {
640                 return PacketTypes.PollDataResponse;
641             }
642             else if (ConnectIndicationEvent.IsConnectIndicationEvent(packet))
643             {
644                 return PacketTypes.ConnectionIndicationEvent;
645             }
646             else if (Refuse.IsRefuse(packet))
647             {
```

```
648                 return PacketTypes.Refuse;
649             }
650             else if (AssociationResponse.IsAssociationResponse(packet))
651             {
652                 return PacketTypes.AssociationResponse;
653             }
654             else if (MdsCreateEvent.IsMdsCreateEvent(packet))
655             {
656                 return PacketTypes.MdsCreateEvent;
657             }
658             else if (AssociationAbort.IsAssociationAbort(packet))
659             {
660                 return PacketTypes.AssociationAbort;
661             }
662             else if (ReleaseRequest.IsReleaseRequest(packet))
663             {
664                 return PacketTypes.ReleaseRequest;
665             }
666             else if (ReleaseResponse.IsReleaseResponse(packet))
667             {
668                 return PacketTypes.ReleaseResponse;
669             }
670             else if (RemoteOperationError.IsRemoteOperationError(packet))
671             {
672                 return PacketTypes.RemoteOperationError;
673             }
674 
675             return PacketTypes.Unknown;
676         }
677 
678         /// <summary>
679         /// Handles network communication.
680         /// </summary>
681         private void nbw_DoWork(object sender, DoWorkEventArgs e)
682         {
683             while (!_networkBGW.CancellationPending)
684             {
685                 // Poll the network
686                 PollNetwork();
687                 System.Threading.Thread.Sleep(1);
688             }
689         }
690 
691         /// <summary>
692         /// Handles reporting changes in connection progress.
693         /// </summary>
694         private void nbw_ProgressChanged(object sender, ProgressChangedEventArgs e)
```

```
695         {
696             Output(e.UserState.ToString());
697         }
698 
699         /// <summary>
700         /// Outputs (message).
701         /// </summary>
702         private void Output(string message)
703         {
704             OutputTextChanged(this, new OutputTextChangedEventArgs(message));
705             OutputText = message;
706             Console.WriteLine(message);
707         }
708 
709         /// <summary>
710         /// Handles the receiving of packets from the Intellivue Monitor after
711         /// the Connection Indication Event is successfully retrieved.
712         /// </summary>
713         private void PollNetwork()
714         {
715             byte[] packet = new byte[Constants.MaximumPacketLength];
716 
717             try
718             {
719                 // Receive a packet from the Intellivue monitor
720                 IPEndPoint externalIpEndPoint = new IPEndPoint(IPAddress.Parse(Constants.IntellivueMonitorHostName),
721                     _listenPort);
722                 packet = _uc.Receive(ref externalIpEndPoint);
723 
724                 // Determine packet type
725                 PacketTypes pt = GetPacketType(packet);
726 
727                 // Handle packet by type
728                 switch (pt)
729                 {
730                     case PacketTypes.AssociationResponse:
731                         // Receiving an assocation response packet
732                         AssociationResponse ar = new AssociationResponse(packet);
733                         Output("Association Response received.");
734                         if (ar.ProtocolVersion != AssociationRequest.ProtocolVersion)
735                         {
736                             Output("Protocol Version mismatch with monitor.");
737                         }
738                         break;
739                     case PacketTypes.MdsCreateEvent:
740                         // Send MDS Create Result as a MDS Create Event was received.
741                         MdsCreateResult mcr = new MdsCreateResult();
```

```
742                     _uc.Send(mcr.ToByteArray(), mcr.Length,
743                             new IPEndPoint(IPAddress.Parse(Constants.IntellivueMonitorHostName), _listenPort));
744                     Output("MDS_Create_Event_received.___Sending_MDS_Create_Result.");
745
746                     // Start timer for requesting data
747                     _timer.Start();
748                     break;
749                 case PacketTypes.Refuse:
750                     // A Refuse message was received, so we need to disconnect.
751                     Shutdown(false);
752                     Output("A_Refuse_message_was_received.");
753                     return;
754                 case PacketTypes.AssociationAbort:
755                     // An Association Abort message was received, perhaps this connection timed out
756                     Shutdown(false);
757                     Output("An_Association_Abort_message_was_received.");
758                     return;
759                 case PacketTypes.PollDataResponse:
760                     if (Status == IntellivueBridgeStatuses.Connected)
761                     {
762                         // Data received... parse data and trigger appropriate events;
763                         PollDataResponse pdr = new PollDataResponse(packet);
764                         HandlePollDataResponse(pdr);
765                     }
766                     break;
767                 case PacketTypes.ReleaseRequest:
768                     // Send a ReleaseResponse if received a ReleaseRequest from the Intellivue
769                     ReleaseResponse rr = new ReleaseResponse();
770                     _uc.Send(rr.ToByteArray(), rr.Length,
771                             new IPEndPoint(IPAddress.Parse(Constants.IntellivueMonitorHostName), _listenPort));
772
773                     Output("Release_request_received,_sending_Release_Response.");
774
775                     // Disconnect
776                     Shutdown(false);
777                     return;
778                 case PacketTypes.ReleaseResponse:
779                     // We must have already sent a Release Request and now received our response
780                     Output("Release_response_received.");
781                     Shutdown(false);
782                     return;
783                 case PacketTypes.RemoteOperationError:
784                     // We received on error... report type
785                     RemoteOperationError error = new RemoteOperationError(packet);
786                     Output("Received_Remote_Operation_Error_of_type:_" + error.ErrorType);
787                     break;
788                 case PacketTypes.AssociationRequest:
```

```
789                     case PacketTypes.ConnectionIndicationEvent:
790                     case PacketTypes.Unknown:
791                         // We should not have received a packet of this type...
792                         Output(String.Format("Unexpected packet received of type {0}.", pt));
793                         break;
794                 }
795             }
796             catch (SocketException soe)
797             {
798                 if (soe.SocketErrorCode == SocketError.TimedOut)
799                 {
800                     // Our connection timed out
801                     Output("Error: Socket connection timed out.");
802                     Shutdown(false);
803                     return;
804                 }
805                 else
806                 {
807                     Output("Error: Socket Exception occurred.  Disconnecting...");
808                     Shutdown(false);
809                     return;
810                 }
811             }
812 
813         }
814 
815         /// <summary>
816         /// Handles the Release Request/Release Response from the monitor
817         /// </summary>
818         private void ReleaseFromMonitor()
819         {
820             if (_uc != null && _networkBGW.IsBusy)
821             {
822                 Output("Sending Release Request...");
823                 ReleaseRequest rreq = new ReleaseRequest();
824                 _uc.Send(rreq.ToByteArray(), rreq.Length,
825                                 new IPEndPoint(IPAddress.Parse(Constants.IntellivueMonitorHostName), _listenPort));
826             }
827             else
828             {
829                 Output("Error: Attempted to send a Release Request but could not.");
830                 Shutdown(false);
831             }
832         }
833 
834         /// <summary>
835         /// Sends disconnect data.
```

```
836         /// </summary>
837         private void SendDisconnectData()
838         {
839             SetDefaultProperties();
840             int ret = Constants.IntValueUndefined;
841             float retf = Constants.FloatValueUndefined;
842             AirwayMacConcentrationEtReceived(this, new FloatValueReceivedEventArgs(ret));
843             AirwayMacConcentrationInspReceived(this, new FloatValueReceivedEventArgs(ret));
844             ArterialBloodPressureReceived(this, new BloodPressureReceivedEventArgs(ret, ret, ret));
845             BisReceived(this, new BisReceivedEventArgs(ret));
846             CO2ETReceived(this, new FloatValueReceivedEventArgs(ret));
847             EmgReceived(this, new EmgReceivedEventArgs(ret));
848             FiO2Received(this, new IntegerValueReceivedEventArgs(ret));
849             HeartRateReceived(this, new HeartRateReceivedEventArgs(ret));
850             ieDESReceived(this, new IntegerValueReceivedEventArgs(ret));
851             IERatioReceived(this, new IntegerValueReceivedEventArgs(ret));
852             ieSEVReceived(this, new IntegerValueReceivedEventArgs(ret));
853             MnAwPReceived(this, new IntegerValueReceivedEventArgs(ret));
854             HematocritReceived(this, new IntegerValueReceivedEventArgs(ret));
855             HemoglobinReceived(this, new IntegerValueReceivedEventArgs(ret));
856             NonInvasiveBloodPressureReceived(this, new BloodPressureReceivedEventArgs(ret, ret, ret));
857             OxygenConsumptionVO2Received(this, new IntegerValueReceivedEventArgs(ret));
858             PerfusionIndicatorReceived(this, new FloatValueReceivedEventArgs(retf));
859             PlateauPressureReceived(this, new IntegerValueReceivedEventArgs(ret));
860             PositiveEndExpiratoryPressureReceived(this, new IntegerValueReceivedEventArgs(ret));
861             PositiveInspiratoryPressureReceived(this, new IntegerValueReceivedEventArgs(ret));
862             PulseFromNonInvasiveBloodPressureReceived(this, new PulseReceivedEventArgs(ret));
863             PulsePlethReceived(this, new PulseReceivedEventArgs(ret));
864             RespiratoryRateReceived(this, new RespiratoryRateReceivedEventArgs(retf));
865             SpO2Received(this, new SpO2ReceivedEventArgs(retf));
866             SqiReceived(this, new SqiReceivedEventArgs(ret));
867             SuppressionRatioReceived(this, new IntegerValueReceivedEventArgs(ret));
868             TemperatureUnspecifiedReceived(this, new TemperatureReceivedEventArgs(retf, Units.DegreesCelsius));
869             TidalVolumeReceived(this, new IntegerValueReceivedEventArgs(ret));
870             TidalVolumeInspiredReceived(this, new IntegerValueReceivedEventArgs(ret));
871         }
872
873         /// <summary>
874         /// Sets the default value of all the properties of the Intellivue Bridge.
875         /// </summary>
876         private void SetDefaultProperties()
877         {
878             // Set all measurements to their undefined values to let the client know that
879             // we are no longer connected to these sensors.
880             int ret = Constants.IntValueUndefined;
881             float retf = Constants.FloatValueUndefined;
882             this.AirwayMacConcentrationET = retf;
```

```
883             this.AirwayMacConcentrationInsp = retf;
884             this.ArterialDiastolic = ret;
885             this.ArterialMAP = ret;
886             this.ArterialSystolic = ret;
887             this.BIS = ret;
888             this.CO2ET = ret;
889             this.EMG = ret;
890             this.FiO2 = ret;
891             this.HeartRate = ret;
892             this.ieDES = ret;
893             this.IERatio = ret;
894             this.ieSEV = ret;
895             this.MnAwP = ret;
896             this.Hematocrit = ret;
897             this.Hemoglobin = ret;
898             this.NonInvasiveDiastolic = ret;
899             this.NonInvasiveMAP = ret;
900             this.NonInvasiveSystolic = ret;
901             this.OxygenConsumptionVO2 = ret;
902             this.PerfusionIndicator = retf;
903             this.PlateauPressure = ret;
904             this.PositiveEndExpiratoryPressure = ret;
905             this.PositiveInspiratoryPressure = ret;
906             this.PulseFromNBP = ret;
907             this.PulsePleth = ret;
908             this.RespiratoryRate = retf;
909             this.SpO2 = retf;
910             this.SQI = ret;
911             this.SuppressionRatio = ret;
912             this.TemperatureUnspecified = retf;
913             this.TemperatureUnspecifiedUnits = Units.DegreesCelsius;
914             this.TidalVolume = ret;
915             this.TidalVolumeInspired = ret;
916         }
917
918         /// <summary>
919         /// Starts the Intellivue Bridge.
920         /// </summary>
921         public void Start()
922         {
923             if (Status == IntellivueBridgeStatuses.Disconnected)
924             {
925                 // Begin connecting to monitor
926                 Status = IntellivueBridgeStatuses.Connecting;
927
928                 try
929                 {
```

```
930                     // Start BootP Server.
931                     Output("Starting BootP Server...");
932                     _bootPServer.Start();
933                 }
934                 catch (SocketException)
935                 {
936                     _connBGW.CancelAsync();
937                     Output("Error: Socket exception occurred during BootP Server commencement.
938 ___________________________________Check the network cable and try again.");
939                     Status = IntellivueBridgeStatuses.Disconnected;
940                 }
941                 catch (Exception ex)
942                 {
943                     _connBGW.CancelAsync();
944                     Output("Error: " + ex.Message);
945                     Status = IntellivueBridgeStatuses.Disconnected;
946                 }
947             }
948             else
949             {
950                 Output("Call to Start discarded as the Intellivue Bridge is already running.");
951             }
952         }
953 
954         void bootPServer_StatusChanged(object sender, BootPServer.StatusChangedEventArgs args)
955         {
956             if (args.Status == BootPServer.BootPServerStatuses.Complete)
957             {
958                 // Start connection monitor for waiting for CIE
959                 _connBGW.RunWorkerAsync();
960 
961                 Output("Started Intellivue Bridge. Waiting for Connect Indication Event.");
962             }
963             else if (args.Status == BootPServer.BootPServerStatuses.NotRunning)
964             {
965                 // BootP server failed
966                 Status = IntellivueBridgeStatuses.Disconnected;
967             }
968         }
969 
970         /// <summary>
971         /// Handles reporting of output from the BootP server.
972         /// </summary>
973         void bootPServer_OutputTextChanged(object sender, BootPServer.OutputTextChangedEventArgs args)
974         {
975             Output(args.OutputText);
976         }
```

```
977
978         /// <summary>
979         /// Shuts down the connection with the Intellivue Monitor (if one exists).
980         /// </summary>
981         public void Shutdown()
982         {
983             if (Status == IntellivueBridgeStatuses.Connected)
984             {
985                 // Shut down bridge
986                 Shutdown(true);
987             }
988             else if (_bootPServer.Status == BootPServer.BootPServerStatuses.Running)
989             {
990                 // Cancel BootP Server
991                 _bootPServer.Cancel();
992             }
993             else
994             {
995                 Output("Shutdown call ignored as this call is valid only when the Intellivue Bridge is Connected.");
996             }
997         }
998
999         /// <summary>
1000        /// Shuts down the connection with the Intellivue Monitor (if one exists).
1001        /// </summary>
1002        private void Shutdown(bool sendReleaseRequest)
1003        {
1004            if (Status == IntellivueBridgeStatuses.Connected || Status == IntellivueBridgeStatuses.Disconnecting
1005                || Status == IntellivueBridgeStatuses.Connecting)
1006            {
1007                Status = IntellivueBridgeStatuses.Disconnecting;
1008
1009                if (_connBGW != null && _connBGW.IsBusy)
1010                {
1011                    // Cancel background worker for receiving Connection Indication Event
1012                    _connBGW.CancelAsync();
1013                }
1014
1015                if (_timer != null)
1016                {
1017                    // Stop the timer so that we no longer request data from the monitor
1018                    _timer.Stop();
1019                }
1020
1021                if (_networkBGW != null && _networkBGW.IsBusy)
1022                {
1023                    if (sendReleaseRequest)
```

```
1024                    {
1025                        // Send a release request
1026                        ReleaseFromMonitor();
1027                        return;
1028                    }
1029
1030                    // Cancel background worker for network communication
1031                    _networkBGW.CancelAsync();
1032                }
1033
1034                if (_uc != null)
1035                {
1036                    // Close the socket
1037                    _uc.Close();
1038                }
1039
1040                _listenPort = -1;
1041
1042                // Set status to disconnected
1043                Status = IntellivueBridgeStatuses.Disconnected;
1044
1045                Output("Intellivue Bridge disconnected.");
1046
1047                // Create UDP client for receiving connect indication event
1048                _uc = new UdpClient(new IPEndPoint(IPAddress.Parse(Constants.InterfaceIP),
1049                                    ConnectIndicationEvent.ListenPort));
1050                _uc.Client.ReceiveTimeout = Constants.ConnectionIndicationEventTimeout;
1051                _uc.Client.SendTimeout = Constants.ConnectionIndicationEventTimeout;
1052
1053                // We are now disconnected, make sure disconnect values are displayed
1054                SendDisconnectData();
1055            }
1056        }
1057
1058        /// <summary>
1059        /// Handles the period sending of data requests.
1060        /// </summary>
1061        private void timer_Elapsed(object sender, System.Timers.ElapsedEventArgs e)
1062        {
1063            if (_uc != null && Status == IntellivueBridgeStatuses.Connected)
1064            {
1065                // Send Poll Data Request
1066                PollDataRequest pdr = new PollDataRequest(_pollNumber);
1067                _uc.Send(pdr.ToByteArray(), pdr.Length,
1068                            new IPEndPoint(IPAddress.Parse(Constants.IntellivueMonitorHostName), _listenPort));
1069                _pollNumber++;
1070            }
```

```
1071                }
1072            }
1073    }
```