An accessible, hands-on tutorial system for image-guided therapy and medical robotics using a robot and open-source software

Release 0.00

Danielle F. Pace[1], Ron Kikinis[1] and Nobuhiko Hata[1]

September 1, 2007

[1]Surgical Planning Laboratory, Brigham and Women's Hospital and Harvard Medical School
Boston, MA, USA

Abstract

This paper describes a new open-source tutorial for image-guided therapy (IGT) and medical robotics that is both accessible and hands-on using the LEGO Mindstorms NXT (a commercially available robotics kit) and 3D Slicer (an open-source application for medical image processing). The tutorial covers all stages of a typical IGT or medical robotics procedure, including the concepts of imaging, preoperative planning, targeting and tracking, navigation and registration, by using the LEGO robot to perform a "needle biopsy" on a phantom (anatomical model) made of traditional LEGO pieces. In addition, this paper describes a C++ library that allows direct control of a LEGO Mindstorms NXT robot from a Linux computer over a USB connection.

Contents

1 Introduction

Image-guided therapy (IGT) and medical robotics compose a modern interventional approach that uses technology to enable new minimally invasive therapies, improve postoperative outcomes, increase the quality and speed of interventional procedures, shorten hospital stays and decrease long-term hospital costs. It is because of these notable advantages that IGT and medical robotics have gained significant interests in research, clinics and industry and have made major advances in recent years [1]. Open-source software projects such as ITK [2], VTK [3] and IGSTK [4] are major contributors to this growth, as are multi-center collaborations such as those described by Hata *et al.* in [5]. The current expansion of IGT and medical robotics means that both academia and industry require specialized personnel in order to continue the forward momentum. Therefore educational materials that both attract people to IGT and medical robotics and provide them with the necessary theoretical knowledge and technical skills must be widely distributed. In order to reach as many people as possible, these tools must be open-source, low-cost and widely available for purchase. This goal has been partially met by tutorials, workshops and seminars associated with the aforementioned open-source software packages [6].

However, current educational tools for IGT and medical robotics are insufficient because the field's intrinsic reliance on equipment means that tutorials must be hands-on in order to be truly effective. Since tracking devices are expensive and phantoms (anatomical models) are time-consuming to construct, this "hands-on" requirement directly conflicts with the equally crucial requirements for educational materials to be open-source, low-cost and widely available. Thus the vast majority of tutorials on IGT and medical robotics are unfortunately not hands-on. An exception is the comprehensive practical tutorial system provided in [7], however it is not accessible because it requires the purchase of a tracking device. To the best of our knowledge, there does not exist a sub-$500 USD, practical tutorial system for IGT and medical robotics available today. Instead, the only options available for many newcomers to the field are to merely read about the subject or, if they are lucky, attend a brief conference workshop that may or may not have a practical component. These educational experiences do not come close to conveying the dynamic and exciting world of IGT and medical robotics: without practical and accessible educational tools, the field of image-guided therapy and medical robotics will fail to capture the best and brightest minds that it needs to continue its rapid growth.

The objective of this paper is to describe a tutorial for image-guided therapy and medical robotics that is hands-on while remaining open-source and accessible. The tutorial uses the LEGO Mindstorms NXT, an inexpensive and widely available robotics kit. The software architecture chosen for this project was 3D Slicer (Massachusetts Institute of Technology Artificial Intelligence Lab and Brigham and Women's

Hospital, Boston, Massachusetts), a comprehensive open-source software package for medical image processing and image-guided therapy [8]. 3D Slicer was selected for its very unrestrictive license and its extensive built-in image processing and IGT functionality. The LEGO robot and 3D Slicer are used to simulate a needle biopsy with a preoperative target (such as that which may be done in potential cases of breast or prostate cancer) on a phantom created out of standard LEGO pieces. In this way, tutorial participants work with both hardware and software while being exposed to all of the typical steps of an IGT or medical robotics procedure, including imaging, preoperative planning, targeting and tracking, navigation and registration.

2 Materials and Methods

2.1 The tutorial supplies

Table 1 lists the materials used in our image-guided therapy and medical robotics tutorial. Tutorial participants are required to supply a LEGO Mindstorms NXT kit, a LEGO Deluxe Brick Box of traditional LEGO pieces, various other small components needed to build the phantom, and a Linux computer with root access. We provide open-source tutorial software in the form of a specialized module in 3D Slicer v. 3, a high-resolution CT volume of the phantom (scanned at 3 mm with no overlapping slices), tutorial slides, assembly instructions for the LEGO robot and the phantom, and a "Phantom Placement Guide" that aids in positioning the LEGO robot and the phantom during the tutorial. Before beginning the tutorial, participants use our provided instructions (created using Lego Digital Designer v. 2 [9]) and the LEGO Mindstorms NXT kit to construct the tutorial robot that will perform the simulated needle biopsy. Users also build the phantom (anatomical model) using the LEGO Deluxe Brick Box, pom-poms, paper and tape.

2.2 Building a robot using LEGO

The LEGO Mindstorms NXT is a basic robotics kit manufactured and sold by The LEGO Group.

Table 1 The materials used in our image-guided therapy and medical robotics tutorial. The left column lists materials that must be supplied by users of the tutorial, while the right column lists open-source materials that we provide as part of the tutorial package.

Materials provided by the tutorial participant	Materials provided as part of the tutorial package
• One LEGO Mindstorms NXT robotics kit (Item #8527) • One LEGO Deluxe Brick Box (Item #6167) • Two pom-poms of diameter 2.5 cm (1 inch) • Two pieces of paper and tape • A Linux computer with root access	• Specialized tutorial module in 3D Slicer v. 3 • CT volume of the phantom • Tutorial slides containing background information and instructions on how to follow the tutorial • Assembly instructions for the LEGO robot and the phantom • Phantom placement guide

Although originally commercialized based on research by the MIT Media Lab [10] as a toy for children, the potential of the LEGO robot as a tool for research and education was quickly taken advantage of by the scientific community. LEGO robots have been used to investigate efficiency and self-organized task allocation in prey retrieval in swarm robotics [11] and to rapidly create mechanical mock-ups of interactive robots in order to elicit user requirements [12]. In the educational domain, LEGO robots have been integrated into undergraduate computer science courses in artificial intelligence [13, 14] while LEGO robot competitions provide popular introductions for children to robotics [15, 16]. In particular, our IGT and medical robotics tutorial was inspired by the work of the Computer-Integrated Surgery Student Research Society (CISSRS), which runs weekend-long CISSRS Surgical LEGO Robot Competitions to expose high-school students to *medical* robotics in particular [17]. The ease with which robots can be quickly constructed and programmed, as well as the extensive open-source software and documentation provided by both The LEGO Group and the extremely active LEGO Mindstorms NXT community, makes LEGO an ideal system with which to build a tutorial for IGT and medical robotics.

Detailed hardware and software specifications of the LEGO Mindstorms NXT can be found in The LEGO Group's description of the product [18] and in several books on the system (such as [19]). Provided in the LEGO Mindstorms NXT kit is the "NXT intelligent brick", containing a 32-bit ARM7 microcontroller, a 8-bit AVR microcontroller, 256 + 4 Kbytes of FLASH memory and 64 Kbytes + 512 bytes of RAM. Also included are four sensors: the touch sensor provides button-press functionality, the sound sensor detects decibels up to 90 dB, the light sensor measures light intensity and the ultrasonic sensor detects objects by measuring the distance to the nearest object in front of it (in units of one centimeter). Three servo motors each have an adjustable power setting to provide rotational movement and a built-in rotation sensor that returns the number of degrees that its motor has rotated through since the sensor's last reset. Finally, 519 LEGO TECHNIC pieces form the basic construction materials used to build a robot. LEGO Mindstorms NXT robots are typically smaller than 2' x 2' x 2' although this obviously depends on the design. Many scientific and educational projects using LEGO Mindstorms use multiple kits to build a single robot, however we constrained ourselves to a single kit in order to minimize cost to the user and therefore satisfy our goal of accessibility.

Typically, when programming a LEGO Mindstorms NXT one writes and compiles the code on a personal computer and then transfers the executable to the robot's intelligent brick using its USB 2.0 or Bluetooth capability. The robot then acts autonomously from the computer when the program is run on the brick. Although LEGO ships its own graphical programming software with the product, most users with programming experience prefer to use an alternative open-source language that is text-based. Of particular note is Not eXactly C (NXC) [20], a C-like language built on top of the affiliated Next Byte Codes (NBC) compiler, and Bricx Command Center [21], a popular integrated development environment (IDE). Both the official graphical LEGO Mindstorms NXT programming software and Bricx Command Center run on Windows and Macintosh systems only. However, UNIX users can write NXC code in any text editor, compile using the command line NBC compiler and transfer the resulting executable to the robot using the open-source LiNXT [22]. Finally, The LEGO Group also provides documentation for advanced users to send direct commands covering all robotic functions from a personal computer to the NXT brick, to create their own sensors, and more [23].

2.3 Robotic control using 3D Slicer

In contrast to the typical programming procedure described above, in our IGT and medical robotics tutorial the LEGO robot does not act autonomously. Instead, all visualization, preoperative planning, robotic control, registration and miscellaneous computation is performed using a specialized 3D Slicer v.

3 module that participants download, build and install. Thus this tutorial builds upon the work done by CISSRS by 1) integrating an advanced image processing and IGT software package currently used for research and development and 2) covering more advanced topics such as preoperative planning and registration.

Our need to send direct commands to the LEGO robot from 3D Slicer led to the development of a C++ robotic control library that enables the control of an LEGO Mindstorms NXT robot from a computer over a USB connection. In particular, one can read sensor values, move the motors and read from the motors' rotation sensors within any C++ program that includes our library. The USB functionality of the LEGO NXT intelligent brick was chosen over Bluetooth to maximize the reliability of the connection, especially in a classroom setting where multiple robots may be in use at the same time. Please see Appendices A and B for a list of the functions provided by our C++ library and a simple example of their use.

Our C++ library was developed based on the open-source projects NXT++ [24] and Device::USB [25]. Although NXT++ provides the same functionality as our library, it was not used in this project because it unfortunately did not work properly on our 64-bit Linux (Fedora Core 5) platform. Our robotic control library also uses the open-source library libusb [26] to access the USB device. At present our robotic control library is available only for the Linux platform, although support for Windows and Macintosh is planned for the near future. Please note that programs using our library should be executed as root, since root access guarantees the permissions required to access the USB device using libusb.

2.4 The tutorial task: a simulated needle biopsy

As previously mentioned, our tutorial uses a needle biopsy as a model of a typical IGT or medical robotics intervention. Equipped with a "needle" that can move up and down, the tutorial robot is used to "biopsy" a "tumour" target on the phantom by aiming the end of the needle actuator at the target (Figure 1).

The LEGO robot designed for our tutorial is shown in Figure 1A. All three motors provided with the LEGO Mindstorms NXT kit are utilized in the design: one to swing the robotic arm from side to side (motor A), one to move the robotic arm forwards/down and backwards/up (motor B), and one to move the needle up and down (motor C). In addition, one of the four LEGO sensors is used: the ultrasonic sensor measures the distance to the nearest object in front of it and is used to get fiducial coordinates on the phantom in the registration process (as described extensively in Section 2.6).

Figure 1B shows a photograph of the phantom created for the IGT and medical robotics tutorial. Using LEGO pieces to create the phantom gave us maximum design flexibility while maintaining affordability for the user. In addition, the dimensions of LEGO pieces are so exceptionally precise that we were able to define a coordinate system in the robot's physical space, known as the patient (robot) coordinate system, or *PCS*, in "LEGO units" of length 0.8 cm: the distance between the centers of adjacent LEGO studs. A red and a green pom-pom (small fuzzy balls typically used in arts-and-crafts projects) are visible in Figure 1B and are referred to as such in the remainder of this paper. The centers of these pom-poms are the two tumour targets. The phantom is composed of a central box containing two smaller boxes into which the pom-pom tumour targets are placed. Four "registration pillars" surround the central box (two tall ones at the back of the phantom and two shorter ones in front). The registration pillars and the ultrasonic sensor are used to find fiducial coordinates in the *PCS* in the registration process. The top halves of the registration pillars are wrapped with paper that is secured with tape so that they can be better

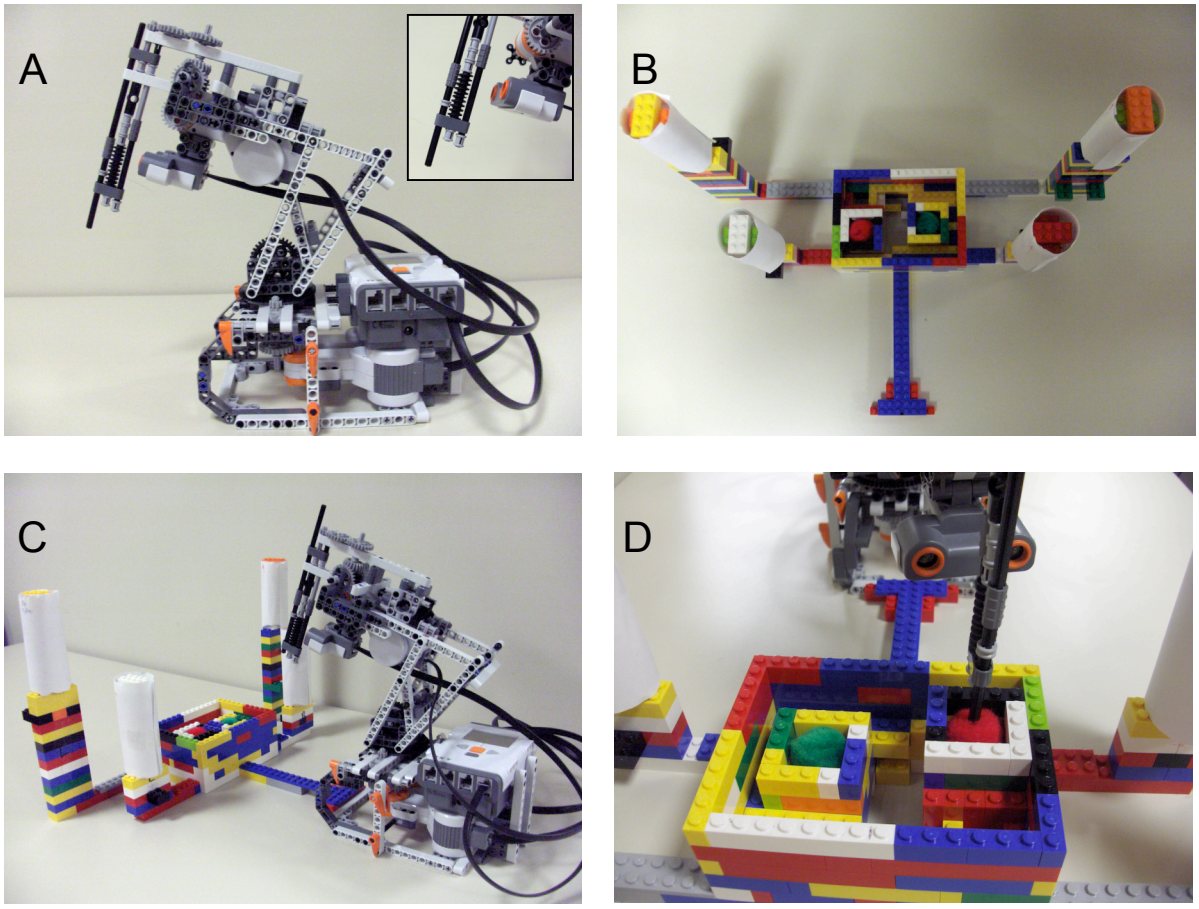

Figure 1 The tutorial task is to have the LEGO robot perform a "needle biopsy" on the phantom. A) The tutorial robot (the inset highlights the needle mechanism and the ultrasonic sensor); B) The phantom (note the red and green pom-pom targets and the registration pillars); C) An example of the tutorial setup; D) A successful needle biopsy of the red pom-pom target.

"seen" by the ultrasonic sensor. Finally, a T-shaped spacer at the bottom of the phantom aids with its positioning during the hands-on components of the tutorial.

Our tutorial is composed of three sections. The Background and Materials section provides theoretical information about image-guided therapy and medical robotics, with a focus on the five tutorial concepts that we aim to cover: imaging, preoperative planning, targeting and tracking, navigation and registration. The next two sections cover the material in a hands-on fashion using the LEGO robot and the phantom (see Table 2). The Basic Tutorial is the first of the two hands-on tutorial sections and provides education on all but the last of the tutorial topics listed above, while the Advanced Tutorial reinforces the concepts of imaging, preoperative planning, targeting and tracking and navigation and also incorporates a registration procedure. In total, the estimated time to complete all three sections is approximately three hours: one hour to build the tutorial robot, one hour to read the material in the Background and Materials section and one hour to complete the hands-on Basic and Advanced tutorial sections.

Table 2 The five tutorial concepts to be covered in a hands-on manner in the basic and advanced tutorial sections.

Tutorial concept	Basic	Advanced	How?
Imaging	✓	✓	CT volume of the phantom
Preoperative planning	✓	✓	Target selection on the CT volume by clicking
Targeting and tracking	✓	✓	Robotic control by the 3D Slicer tutorial module
Navigation	✓	✓	Report of the final needle position after the biopsy is executed
Registration		✓	Using eight fiducials in the image and the patient (robot) coordinate systems and a rigid landmark registration algorithm

2.5 The basic tutorial covers the typical IGT and medical robotics workflow

In the basic tutorial, participants use the spacer at the base of the phantom and the phantom placement guide to put the phantom in a predefined position and orientation with respect to the LEGO robot (see Figure 2A). The LEGO robot is also initially positioned such that the robotic arm is centered and pushed as far back as possible, and the needle is in its highest position. We must specify the robot's initial configuration because there is no way to automatically sense it at the beginning of the tutorial. The user then follows the tutorial slides to move through the typical image-guided therapy / medical robotics workflow. He or she will upload the CT volume of the phantom in 3D Slicer and establish the USB connection between the LEGO robot and the 3D Slicer tutorial module. The user then selects the target coordinate in the image coordinate system, or *ICS*, by clicking on the appropriate point in the CT image volume (see Figure 3). Typically this target coordinate will be the center of one of the pom-poms on the phantom, however the user is free to try to have the robot's needle reach additional targets if desired.

Once the target has been defined in the *ICS* of the CT volume, the user can instruct the LEGO robot to "execute the biopsy", that is, to have the robotic arm and needle move so that the needle tip reaches the target. Since the physical relationship between the LEGO robot and the phantom is predefined, the rigid transformation that transforms image coordinates into coordinates in the patient (robot) coordinate system within which the robot operates is approximately constant for all executions of the basic tutorial. The target coordinate in the *ICS* is therefore transformed into a target coordinate in the *PCS* by multiplying it by a hard-coded registration matrix. This registration matrix was the result of running a rigid landmark registration algorithm using corresponding pairs of image and patient (robot) coordinates from 72 fiducial points distributed over the phantom as input. Thus in the basic tutorial no registration procedure is explicitly performed by the user.

Finally, in order to "execute the biopsy" and move the robot's needle to the target coordinate in the *PCS* we need to calculate the number of rotations that each of the three motors must execute (i.e. we must solve the robot's inverse kinematics problem). We identified three key angles on the robot's frame and

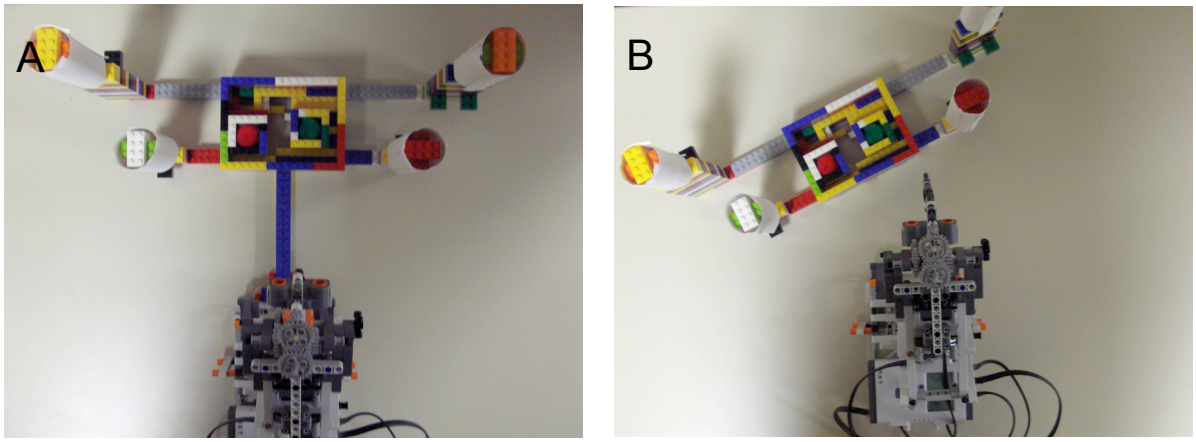

Figure 2 Placement of the phantom relative to the LEGO robot for the A) basic tutorial; B) advanced tutorial.

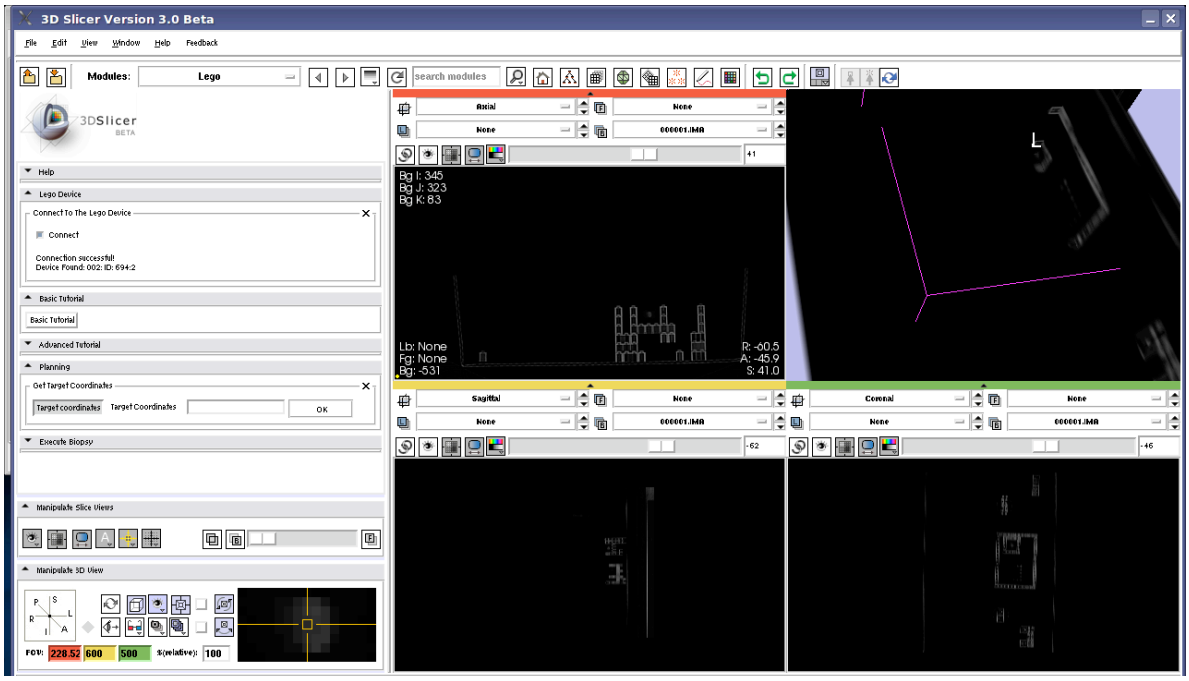

Figure 3 Screenshot of the 3D Slicer tutorial module during the preoperative planning (target selection) phase. The user selects the target by clicking on the appropriate point in the CT volume. Note that both the LEGO bricks and the pom-pom targets show up well in the CT volume.

empirically determined interpolating polynomial relationships between these angles and the degrees of rotation of each motor. The angle values that will lead to the needle hitting the target are found geometrically; these angles are then converted into motor rotations using the interpolating polynomial relationships described above. In this section of the tutorial the user physically views the robot's needle reach the target. The target coordinate in the patient (robot) coordinate system is displayed, as is the coordinate in the *PCS* that the needle tip actually reached. The later is calculated by geometrically solving the LEGO robot's forward kinematics problem using the interpolating polynomials and the degrees of motor rotations that were actually executed by each motor (which are close to, but not exactly, the degrees of motor rotation commanded due to slight imprecision within the servo motors).

2.6 The advanced tutorial introduces registration

The advanced tutorial builds upon the concepts demonstrated in the basic tutorial by introducing the topic of registration to the user. The user can place the phantom in any position with respect to the LEGO robot as long as 1) the tip of the robot's needle can reach both pom-poms and 2) the phantom's four registration pillars will be within range of the LEGO robot's ultrasonic sensor as its robotic arm scans from side to side. The phantom placement guide shows one example of how to place the phantom relative to the LEGO robot so that these two conditions are met, however the user is encouraged to repeat the advanced tutorial section with different valid phantom positions.

The steps of the advanced tutorial are equal to those of the basic tutorial with the exception of the additional registration section. The user first loads the CT volume, establishes the connection between the LEGO robot and the 3D Slicer tutorial module, and selects a target coordinate in the image coordinate system by clicking on a point in the CT volume.

Figure 4 illustrates the registration process used. We perform registration between the image coordinate system, or *ICS*, and the patient (robot) coordinate system, or *PCS* (shown in Figure 4A), using a rigid landmark registration algorithm on the *ICS*-*PCS* pairs of coordinates corresponding to the eight fiducial points on the phantom. The locations of these fiducials are shown in Figure 4B: four are centered on the front side at the top surface of each registration pillar, and four are located on the top surface of each corner of the phantom's central box. We followed the suggestions of West *et al.* when choosing the number and location of the fiducials: 1) the fiducial points are in a nonlinear configuration, 2) their center lies close to the two pom-pom targets (the critical regions), 3) the distance between them is relatively large and 4) we use many of them [27].

In theory, one could use the ultrasonic sensor to generate the lengths of the four line segments from the origin of the *PCS* to the closest edge of the registration pillars along with the angles between them and the y-axis of the *PCS*. Together with prior knowledge of the phantom's structure, this information would be sufficient to calculate the fiducial coordinates in the *PCS* regardless of the position and orientation of the phantom with respect to the LEGO robot. Unfortunately though, the accuracy of the ultrasonic sensor (0 to 255 cm ± 3 cm [28]) is not high enough for the required distance measurements to be sufficiently accurate.

Instead, our requirement that the robot's needle must be able to reach both pom-pom targets constrains both the distance *d* from the origin of the *PCS* to the phantom (shown in Figure 4C) and the orientation of the phantom relative to the LEGO robot. If we define α as the angle between the y-axis of the *PCS* and the line perpendicular to the front of the phantom that passes through the origin of the *PCS*, the eight fiducial coordinates in the *PCS* can be calculated using α and prior knowledge of the phantom's structure.

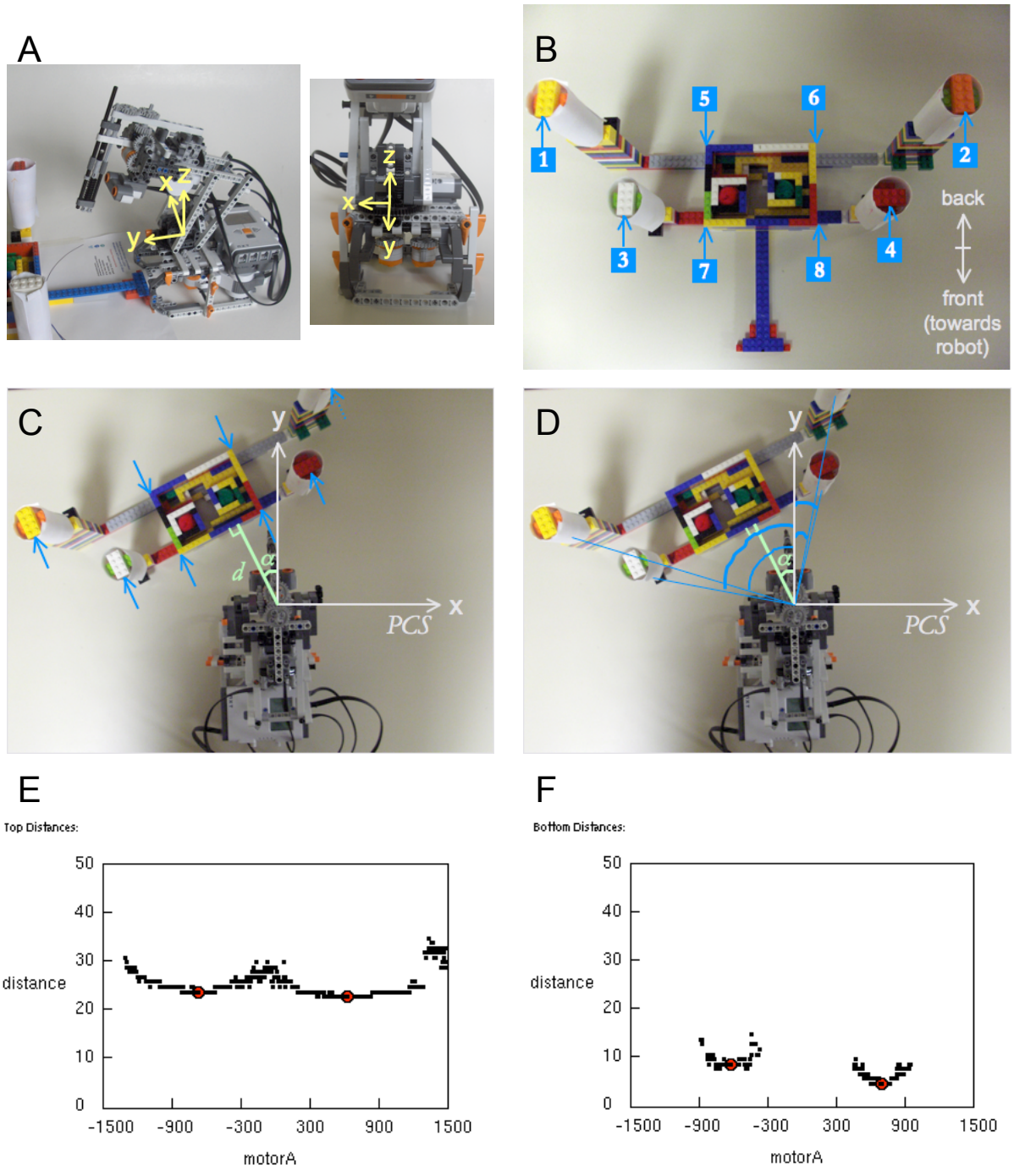

Figure 4 Registration in the advanced tutorial. A) Two views of the patient (robot) coordinate system, or *PCS*; B) The eight fiducial points on the phantom; C) Knowledge of the distance *d* and the angle α are sufficient along with knowledge of the phantom's structure to calculate the eight fiducial coordinates in the *PCS*; D) The average of the four angles between the y-axis of the *PCS* and the line segments from the origin of the *PCS* to the registration pillars approximates α; E, F) Sample distance profiles of *motorA* (the number of degrees through which motor A has turned to move the robotic arm from left to right) versus the distances (cm) returned by the ultrasonic sensor for the top and bottom swipes, respectively. The red dots show the automatically detected local minima.

We approximate this angle α by averaging the four angles between the y-axis of the *PCS* and the line segments from the origin of the *PCS* to the closest edge of the registration pillars (Figure 4D). These four angles are found by using the ultrasonic sensor to generate a distance profile at two different vertical levels: a "top" profile for the tall registration pillars at the back of the phantom and a "bottom" profile for the shorter registration pillars at the front. The robotic arm scans slowly from left to right while continuously polling the ultrasonic sensor for distance measurements and motor A's rotation sensor for the current number of degrees rotated. Figures 4E and 4F show examples of these profiles. Each of the two local minima of each profile corresponds to the number of degrees by which motor A must be rotated so that the center of the ultrasonic sensor faces the associated registration pillar. These *motorA* values can be transformed into the four angles required to calculate α using additional interpolating polynomials that were empirically determined.

Both the top and bottom distance profiles have a characteristic shape that is relatively constant between trials of the advanced tutorial and merely shifts from light to right with different legal positionings of the phantom. It is therefore simple to automatically find the two local minima on each profile. Although a very wide median filter is first used on both scans to reduce noise, a different algorithm is used to find the local minima on the top and bottom profiles in order to take advantage of their different characteristic shapes. Following the median filter, the top profile typically has only two sections that are local minima, that is, there are two stretches in the array where a section of equal distances is preceded and succeeded by sections of larger distances. The middles of these two such sections represent the local minima. If there is only one local minimum (for example if the phantom is positioned at a sharp angle to the robot so that only one and a half curves are present in the top profile), the one local minimum is found as described above and the other is extrapolated using it and the width of the one complete curve. The first and last local minima are chosen if three or more local minima are found, which gives reasonable results in our experience. The bottom profile is even more consistent than the top profile. Two disconnected curves result from filtering the bottom profile with the median filter, and so the middles of these curves give an excellent approximation to the local minima. We do not have to accommodate the possibility of there being one or 3+ curves because this virtually does not happen so long as the user positions the phantom according to the tutorial instructions.

Thus in order to find the fiducial coordinates in the patient (robot) coordinate system the user simply instructs the 3D Slicer tutorial module to scan for them. The two distance profiles are displayed with the automatically detected local minima highlighted, as are the eight calculated fiducial coordinates in the patient (robot) coordinate system. If there was a problem with the scan or with the automatic local minima detection the user has the opportunity to repeat the scan until satisfied with the results.

In the advanced tutorial, tutorial participants select the eight fiducial coordinates in the image coordinate system by clicking on the points in the CT volume of the phantom. The registration matrix is then found using a landmark rigid registration algorithm on the eight pairs of corresponding fiducials in the *ICS* and the *PCS*. Once the target in the *PCS* is determined by multiplying the target in the *ICS* by the registration matrix, the robot "executes" the biopsy in the same way as in the basic tutorial. Displayed to the user following the biopsy is the registration matrix that was used, the target in the *PCS* and the final needle position in the *PCS*.

2.7 Assessment of targeting accuracy

The goal of this project is to create an educational, accessible and practical tutorial, not to create a needle biopsy system with sub-millimeter accuracy. However, targeting accuracy remains important because

continued failure of the robot to hit the target will only frustrate and distract the user. Keeping this in mind, the best way to evaluate targeting accuracy is to determine the percentage of trials in which the LEGO robot's needle reaches the pom-pom target.

There are two sources of error that accumulate to form the *total targeting error*, defined as the Euclidean distance in the patient (robot) coordinate system between the true target on the phantom (the pom-pom center) and the final position of the robot needle tip. First, we define *registration error* as the Euclidean distance in the *PCS* between the true target coordinate and the target in the *PCS*, that is, the coordinate that the robot aims at following registration. Registration error includes the user's error in selecting the target in the image coordinate system as well as the target registration error (TRE), which in our case is the Euclidean distance between the point in the *PCS* that actually corresponds to the user-selected target in the *ICS* and the target in the *ICS* multiplied by the registration matrix (see [27] for additional information about target registration error and its multiple components). Second, we define *execution error* as the distance between the point in the *PCS* that the robot aims for and the final position of the robot needle tip. Execution error comprises inexact initial positioning of the robot, error in the polynomial relationships between robot beam angles and motor rotations (composed of both error in the interpolation and measurement error in the sample points used), differences between the degrees of motor rotation commanded and the degrees executed, and finally any shakiness in the robot's movement.

In order to evaluate the total targeting error of our tutorial system, we purchased a LEGO Mindstorms NXT kit and a LEGO Deluxe Brick Box and built the tutorial robot and phantom described above. For each of the two targets in the image coordinate system corresponding to the centers of the red and green pom-pom targets (as shown in Figure 1B), we completed the basic tutorial five times. The LEGO robot was manually returned to the correct initial position between trials in cases when the robotic arm did not move completely back to the initial position following the biopsy. We considered the target to be "hit" if the final needle position fell into the small 2.4 cm x 2.4 cm x 1.9 cm box surrounding the pom-pom. The coordinates of the actual centers of both pom-pom targets in the *PCS* were determined by counting the number of LEGO units in all three dimensions between the pom-pom centers and the origin of the *PCS*; the accuracy of these true target coordinates is extremely high due to the precision of the LEGO manufacturing process. We recorded the displayed target in the *PCS* (the coordinate in the *PCS* that the robot was aiming for) and determined the final needle position in the *PCS* by visual inspection using the studs on the phantom's LEGO bricks as markers (we estimate that the accuracy of this visual inspection is approximately 0.25 LEGO units, which equals 0.2 cm). The total targeting error, registration error and execution error were calculated as described above, as were the means for the "red" trials, the "green" trials and the total number of trials.

Similarly, we performed an accuracy assessment of the advanced tutorial section using three different phantom positions corresponding to $\alpha = 0^\circ$, $\alpha = -14.5^\circ$ (a typical angle where the phantom is to the left of the robot) and $\alpha = 25^\circ$ (an extreme angle where the phantom is to the right of the robot). Once again, for each phantom position we targeted the centers of the red and green pom-poms in five trials. As in the accuracy assessment of the basic tutorial, we repeatedly used the same target coordinates for the red and green pom-poms in the *ICS* and performed a manual reset of the robotic arm between trials if it did not return exactly to the initial position. We also had the tutorial software output the value of α determined by the registration algorithm specifically for this accuracy assessment. The true coordinates of the centers of the pom-pom targets were calculated algebraically using the true α value, the estimated value of d and knowledge of phantom's structure. The target in the *PCS* after registration was recorded from the display on the screen. Finally, we calculated the final needle position in the *PCS* using the distance in all three dimensions between the needle tip and the center of the pom-pom (once again visually determined with an accuracy of approximately 0.2 cm), the true value of α, the estimated value of d and knowledge of the

phantom's structure. We counted the number of target "hits" and calculated the total targeting error, registration error and execution error as well as the "red", "green" and total means.

3 Results

3.1 The IGT and medical robotics tutorial is highly accessible

Our tutorial is unique because it provides education on advanced topics in image-guided therapy and medical robotics while remaining accessible and hands-on. One of the most important requirements of our tutorial is accessibility, as it will not be widely used if the required materials are not widely available and affordable. All tutorial materials not provided by us are available in both stores and online (the LEGO Mindstorms NXT and Deluxe Brick box are available from [29], pom-poms can be found in craft stores or from [30]). Although future plans to provide the 3D Slicer tutorial module for Windows and Macintosh systems will increase accessibility, most users in university computer science and engineering departments will have access to Linux computers and would therefore be able to use our tutorial. The total cost of our tutorial (using the sources in [29] and [30] and ignoring the negligible cost of paper and tape) is $308.47 USD plus applicable taxes and shipping charges. This cost is well below our goal of $500 USD and represents an affordable cost to university departments. After the initial investment the tutorial can be run an infinite number of times with no cost except for the price of replacement batteries; finally many computer science and engineering departments already own LEGO Mindstorms NXT kits because they themselves run educational programs for younger students. It is clear that the tutorial remains accessible to departments, surgeons and any other individuals interested in image-guided therapy and medical robotics.

3.2 Assessment of targeting accuracy

The results of the accuracy assessment are summarized in Table 3: please recall our previous argument that the percentage of times in which the target is hit is the best indicator of sufficient accuracy, as this is the goal that the user will see. In addition, although total targeting error is composed of registration error and execution error, the sum of registration error and execution error is not expected to equal the total targeting error because execution error may be partially compensated for by registration error in an opposing direction.

Our results show that the basic tutorial is highly reliable: the target was hit in 100% of the trials, while the mean total targeting error is relatively low. The registration error is minor as expected since 72 fiducials were used to create the registration matrix used in the basic tutorial, while the execution error is much larger. Thus the total targeting error in the basic tutorial is primarily due to execution error: we can calculate the goal target with relatively high accuracy, but have difficulty getting the robot's needle there. Errors related to misplacement of the robotic arm initially and differences between the degrees of motor rotation commanded and those executed are expected to be small and are not easily controlled. Therefore the main factors influencing execution error, and therefore total targeting error, are errors in the relationships between beam angles and motor rotations and shakiness of the robot's movement.

100% of the targets were hit in the advanced tutorial for typical values of α, while the percentage of times in which the target was hit decreases with extreme values of α. However, even for $\alpha = 25^{\circ}$ the percentage

Table 3 Results of the accuracy assessment for the basic and advanced tutorials. The total targeting error, registration error and execution error values are Euclidean distances as described in Section 2.7, while the absolute error in α is the difference between the known value of α and that determined by the registration procedure in the advanced tutorial. Means in the "Red target" and "Green target" columns are means over five trials, whereas means in the "Both targets" columns are means over all ten trials. All values are rounded to the nearest tenth of a centimeter or degree.

Basic tutorial:

	Red target	Green target	Both targets
# Times target hit	**5/5**	**5/5**	**10/10**
Mean total targeting error	1.2 cm	1.2 cm	1.2 cm
Registration error	0.2 cm	0.2 cm	0.2 cm
Mean execution error	1.2 cm	1.2 cm	1.2 cm

Advanced tutorial:

	$\alpha = 0^\circ$			$\alpha = -14.5^\circ$			$\alpha = 25^\circ$		
Mean absolute error in α	1.3°			1.3°			1.9°		
	Red target	Green target	Both targets	Red target	Green target	Both targets	Red target	Green target	Both targets
# Times target hit	**5/5**	**5/5**	**10/10**	**5/5**	**5/5**	**10/10**	**3/5**	**4/5**	**7/10**
Mean total targeting error	1.4 cm	1.3cm	1.3 cm	1.2 cm	1.0 cm	1.1 cm	1.6 cm	1.9 cm	1.7 cm
Mean registration error	0.4 cm	0.6 cm	0.5 cm	0.7 cm	0.4 cm	0.6 cm	0.7 cm	0.8 cm	0.8 cm
Mean execution error	1.6 cm	1.2 cm	1.4 cm	1.4 cm	0.9 cm	1.2 cm	1.8 cm	1.9 cm	1.8 cm

of times in which the target was hit was a reasonable 70%, meaning that the user could potentially have to repeat the advanced tutorial a couple of times at most. As expected, registration error in the advanced tutorial is higher than registration error in the basic tutorial. This is due to both the fact that we use eight fiducials rather than 72 and increased fiducial localization error for both the fiducials in the image coordinate system and the patient (robot) coordinate system. Users may not be accurate in clicking on the fiducials in the *ICS*, whereas the fiducials found in the *PCS* are influenced by error in finding the local minima of the *motorA* versus ultrasonic distance curves and in converting the *motorA* values of the detected local minima into robot beam angles. In addition, the distance between the phantom and the origin of the *PCS* might not equal the estimated distance *d* used, the average of the registration pillar angles might not be exactly equal to the true value of α and, even if they could be found exactly, the local minima might not correspond exactly to the registration pillar angles. Yet despite these multiple sources of error, the mean absolute error in α (which encapsulates most of the sources of registration error described above) was in all cases less than two degrees. Finally, as in the basic tutorial, execution error is larger than registration error. Since execution error is the Euclidean distance between the actual position of the target and the coordinate that was aimed for after registration, the differences in mean execution error between the basic and advanced tutorials and between trials of different α values in the advanced tutorial have nothing to do with the quality of the registration. Instead, they are primarily related to the fact that the accuracies of the interpolated motor rotation / robot beam angle relationships and the shakiness of the robotic arm both vary when the robotic arm moves to targets corresponding to different α values (where α in the basic tutorial is 0° by definition).

In passing, please note that the error values shown in Table 3 are not completely accurate. In the basic tutorial data, the total targeting error and the execution error are based on visual inspection of the final needle position and will invariably contain some inaccuracies. However, registration error can be calculated with high accuracy because the actual coordinates of the pom-pom centers and the target that the robot was aiming for in the *PCS* are known. In the advanced tutorial, the actual coordinates of the pom-pom centers are calculated using the estimate for *d* and will therefore have some error. The final needle position is once again based on visual inspection but also has additional error compared to the basic tutorial because the estimate for *d* is used in its calculation. Therefore in the advanced tutorial data, error is now introduced into the measurements of registration error, and error is increased compared to the basic tutorial in the measurements of total targeting error and execution error.

4 Discussion

Currently it is next to impossible for newcomers to IGT and medical robotics to find hands-on educational tools that encapsulate the true nature of the field. Our tutorial on image-guided therapy and medical robotics can be used by anyone with elementary knowledge of algebra to understand the fundamental steps of such procedures in a practical way. We foresee the use of our tutorial as part of labs in undergraduate medical informatics courses, by beginning graduate students in the field, by medical students as an introduction to image-guided therapies and medical robotics, by high-school students in workshops encouraged to increase interest in computer science and engineering, and by various others interested in IGT and medical robotics.

That being said, there is an important fundamental difference in approach between this work and that of other LEGO educators, such as the competitions organized by CISSRS. Whereas the focus of the CISSRS competitions is to get students excited about computer science and engineering, the goal of our tutorial is to provide education on the main steps of a typical IGT or medical robotics procedure. In the

former case creativity in robotic design and in programming are emphasized, while in the current version of our tutorial the robot design is preset and no programming is done by the user. This is not to say that our tutorial system cannot be used in student competitions or should not inspire interest in image-guided therapy and medical robotics. Instead, our tutorial would be an excellent introduction to IGT and medical robotics that students in such competitions could complete before working on their own creations. However, our tutorial has primarily been designed for use by more "serious" students of IGT and medical robotics in a classroom or laboratory setting.

Future work in this project includes enhancing the 3D Slicer tutorial module so that it can be run under Windows and Macintosh systems in addition to the Linux platform. This will increase accessibility and widespread use of our tutorial. In addition, we plan to enhance the navigation provided to the user in the "execute biopsy" phase by providing real-time visualization of the current needle position overlaid onto the CT volume of the phantom. Finally, we would like to encourage more interactivity into the tutorial by having the user play with different targets, registration algorithms, phantom designs and more.

5 Conclusions

In this paper we have shown that comprehensive, hands-on educational materials for image-guided therapy and medical robotics can be built using basic tools and open-source software. This tutorial both uses open-source software packages and will be supplied in an open-source fashion itself in order to meet the currently unsupplied need for accessible and hands-on educational tools in IGT and medical robotics. In addition, we describe a C++ library that can be used to control a LEGO Mindstorms NXT robot from a personal computer.

6 Acknowledgement

This publication was made possible by grant numbers 5U41RR019703, 5P01CA067165 and 5U54EB005149 from NIH. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the NIH. This study was also in part supported by NSF 9731748 and CIMIT. Thanks also to Terry Peters, Ph.D of the Robarts Research Institute and The University of Western Ontario; Steven Canvin of The LEGO Group; Cory Walker, NXT++ developer; G. Wade Johnson, Device::USB developer; Steve Pieper, Ph.D, Haiying Liu, Junichi Tokuda, Ph.D, Christoph Ruetz and Philip Mewes of the Surgical Planning Laboratory; and to the entire LEGO Mindstorms community.

A Functions of our robotic control library

Below is a list of the public functions offered by our C++ library enabling control of a LEGO Mindstorms NXT robot. More extensive documentation on the use of these functions has been included with this submission.

```
// Constructor and destructor
vtkLegoUSB();
~vtkLegoUSB();
```

```
// Open and close the USB connection to the LEGO robot
int OpenLegoUSB();
int CloseLegoUSB();

// Set up the sensors before their use
void SetSensorLight(int port, bool active);
void SetSensorTouch(int port);
void SetSensorSound(int port, bool dba);
void SetSensorUS(int port);
void SetUSOff(int port);
void SetUSSingleShot(int port);
void SetUSContinuous(int port);
void SetUSEventCapture(int port);
void SetUSContinuousInterval(int port, int interval);

// Read from the sensors
int GetLightSensor(int port);
bool GetTouchSensor(int port);
int GetSoundSensor(int port);
int GetUSSensor(int port);

// Move and stop the motors
void SetMotorOn(int port, int power);
void SetMotorOn(int port, int power, int tachoCount);
void MoveMotor(int port, int power, int tachoCount);
void StopMotor(int port, bool brake);

// Read the current degrees of motor rotation from the motors
int GetMotorRotation(int port, bool relative);
void ResetMotorPosition(int port, bool relative);

// Play a tone on the LEGO robot
void PlayTone(int frequency, int duration);

// Get the connection status of the LEGO robot
char * GetStatus();

// Get information about the LEGO robot
char * GetDeviceFilename();
int GetIDVendor();
int GetIDProduct();

// Send direct commands to the LEGO robot - advanced users only
void SendCommand (char * outbuf, int outbufSize, char * inbuf, int inbufSize);
// Get the LS status of the LEGO robot - advanced users only
int LSGetStatus(int port);
```

B Example use of our robotic control library

In order to use our library to control a LEGO Mindstorms NXT robot, simply include `NXT_USB.h`. Installation instructions are provided in the README file included with this submission.

```
#include "NXT_USB.h"

int main (int argc, char* argv[])
```

```
{
    // Create the NXT_USB object
    NXT_USB* legoUSB = new NXT_USB();

    // Open the connection to the LEGO robot
    int open = legoUSB->OpenLegoUSB();

    // Return if we cannot open the connection
    if (open == 0)
    {
        std::cerr << "Could not connect: " << legoUSB->GetStatus() << std::endl;
        return 1;
    }

    // Setup:  the touch sensor should be attached to input port 1
    legoUSB->SetSensorTouch(IN_1);

    // Move the motor attached to output port A (by 180 degrees at +50% power) once
    // the button on the touch sensor has been pressed
    while (!legoUSB->GetTouchSensor(IN_1)) {};
    legoUSB->SetMotorOn(OUT_A, 50, 180);

    // Close the connection to the LEGO robot
    legoUSB->CloseLegoUSB();
    delete legoUSB;

    return 0;
}
```

References

[1] S. DiMaio, T. Kapur, K. Cleary, S. Aylward, P. Kazanzides, K. Vosburgh, R. Ellis, J. Duncan, K. Farahani, H. Lemke, T. Peters, W. Lorensen, D. Gobbi, J. Haller, L. Clarke, S. Pizer, R. Taylor, R. Galloway Jr., G. Fichtinger, N. Hata, K. Lawson, C. Tempany, R. Kikinis, and F. Jolesz. Challenges in Image-Guided Therapy System Design. *NeuroImage*, 37(Suppl 1): S144-S151, 2007.

[2] L. Ibanez and W. Schroeder. *The ITK Software Guide*. Kitware, Inc. ISBN 1-930934-10-6, http://www.itk.org/ItkSoftwareGuide.pdf, 2003.

[3] W. Schroeder, K. Martin, and W. Lorensen. *The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics, 2nd ed*. Kitware, Inc. ISBN 1-930934-12-2. Prentice-Hall, Old Tappan, N.J., 1998.

[4] K. Gary, L. Ibanez, S. Aylward, D. Gobbi, M.B. Blake, and K. Cleary. IGSTK: An Open Source Software Toolkit for Image-Guided Surgery. *Computer*, 39(4):46-53, 2006.

[5] N. Hata, S. Pieper, F. Jolesz, C. Tempany, P. Black, S. Morikawa, H. Iseki, M. Hashizume, and R. Kikinis. Application of Open Source Image Guided Therapy Software in MR-Guided Therapies. In *Proceedings of the 10th International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI'07)*, 2007. To appear.

[6] National Alliance for Medical Image Computing. Training, available at http://wiki.na-mic.org/Wiki/index.php/Training:Main, 2007.

[7] H. Liu and N. Hata. Slicer User Training 101: IGT Edition, available at http://wiki.na-mic.org/Wiki/images/8/89/Igt-tutorial-updated.ppt, 2006.

[8] D. Gering, A. Nabavi, R. Kikinis, N. Hata, L. O'Donnell, W. Eric L. Grimson, F. Jolesz, P. Black, and W. Wells III. An Integrated Visualization System for Surgical Planning and Guidance Using Image Fusion and an Open MR. *Journal of Magnetic Resonance Imaging,* 13(6): 967-975, 2001.

[9] Lego Digital Designer is available for download at http://ldd.lego.com

[10] M. Resnick. Behavior Construction Kits. *Communications of the ACM,* 36(7):64-71, 1993.

[11] T.H. Labella, M. Dorigo, and J.L. Deneubourg. Efficiency and Task Allocation in Prey Retrieval. In *Proceedings of the Frist International Workshop on Biologically Inspired Approaches to Advanced Information Technology*, Lecture Notes in Computer Science, 3141:274-289, 2004.

[12] C. Bartneck and J. Hu. Rapid Prototyping for Interactive Robots. In *Proceedings of the 8th Conference on Intelligent Autonomous Systems (IAS-8)*, 2004.

[13] F. Klassner. A Case Study of LEGO Mindstorms'™ Suitability for Artificial Intelligence and Robotics Courses at the College Level. *ACM SIGCSE Bulletin,* 34(1):8-12, 2002.

[14] A.N. Kumar. Using Robots in an Undergraduate Artificial Intelligence Course: An Experience Report. In *Proceedings of the 31st Annual ASEE/IEEE Frontiers in Education Conference*, 2001.

[15] D. Opplinger. Using First Lego League to Enhance Engineering Education and to Increase the Pool of Future Engineering Students. In *Proceedings of the 32nd Annual ASEE/IEEE Frontiers in Education Conference*, 2002.

[16] H.H. Lund and L. Pagliarini. RoboCup Jr. with LEGO MINDSTORMS. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1(1):813-819, 2000.

[17] O. Gerovich, R.P. Goldberg, and I.D. Donn. From Science Projects to the Engineering Bench. *IEEE Robotics & Automation Magazine*, 10(3):9-12, 2003.

[18] The LEGO Group. NXT Technology Overview, available at http://mindstorms.lego.com/eng/Overview/default.aspx, 2007.

[19] B. Bagnall. *Maximum Lego NXT: Building Robots with Java Brains*. ISBN 0973864915. Variant Press, Winnipeg, M.B., Canada, 2007.

[20] Not eXactly C is available for download at http://bricxcc.sourceforge.net/nbc

[21] Bricx Command Center is available for download at http://bricxcc.sourceforge.net

[22] LiNXT is available for download at http://sourceforge.net/projects/linxt

[23] The LEGO Group. Mindstorms NXT'REME, available at http://mindstorms.lego.com/Overview/NXTreme.aspx, 2007.

[24] NXT++ is available for download at http://nxtpp.sourceforge.net

[25] Device::USB is available for download at http://search.cpan.org/~gwadej/Device-USB-0.21

[26] libusb is available for download at http://libusb.sourceforge.net

[27] J.B. West, M.J. Fitzpatrick, S.A. Toms, C.R Maurer, and R.J. Maciunas. Fiducial Point Placement and the Accuracy of Point-based, Rigid Body Registration. *Neurosurgery*, 48(4):810-817, 2001.

[28] The LEGO Group. LEGO Mindstorms User Guide, 2006 (included with the purchase of a LEGO Mindstorms NXT kit)

[29] The LEGO Mindstorms NXT robotics kit (#8527) and the LEGO Deluxe Brick Box (#6167) are available for purchase at http://shop.lego.com

[30] Pom-poms are available for purchase at http://www.amazon.com/PAC-1859614-Poms-ClassPack-Sizes/dp/B0006HXNRY/ref=sr_1_12/104-9511701-9925562?ie=UTF8&s=office-products&qid=1187791961&sr=8-12