LIBROB: AN AUTONOMOUS ASSISTIVE LIBRARIAN

Costanza Di Veroli, Cao An Le, Thibaud Lemaire, Eliot Makabu, Abdullahi Nur
Vincent Ooi, Jee Yong Park, Federico Sanna
Imperial College London
Kensington, London SW7 2AZ
Email: {cd1915, cal115, tvl18, em3215, an3715, vo115, jyp14, fs2215}@imperial.ac.uk
Supervisor: PROF Y DEMERIS

***Abstract*— This study explores how new robotic systems can help library users efficiently locate the book they require. A survey conducted among Imperial College students has shown an absence of a time-efficient and organised method to find the books they are looking for in the college library. The solution implemented, LIBROB, is an automated assistive robot that gives guidance to the users in finding the book they are searching for in an interactive manner to deliver a more satisfactory experience. LIBROB is able to process a search request either by speech or by text and return a list of relevant books by author, subject or title. Once the user selects the book of interest, LIBROB guides them to the shelf containing the book, then returns to its base station on completion. The robot showed to reduce the time necessary to find a book by 47.4%, and left 80% of the users satisfied with their experience, proving that human-robot interactions can greatly improve the efficiency of basic activities within a library environment.**

I. INTRODUCTION

Libraries have always represented a centre of learning and development since their inception. They play a crucial role in many academic fields but more so in universities where an abundant range of written materials are made available for both students and researchers alike. As such, it is important that students, researchers, and others can easily access all the resources available. One obstacle that many users of the library face is easily locating a specific book title. A survey was conducted recently in which students were asked about the length of time they would spend searching for a book [1]. The results showed that 82% of students spent more than 5 minutes searching for a book. This shows that the process by which books are located can be improved in order give students easier access to the available resources. This paper proposes the use of an autonomous robot to guide the user directly to the book's location in order to save library-user's time. Currently, in order to locate a book, the user must manually type the book title on a computer into the library's search engine, which returns the relevant library floor and the book's identification number. The user is then expected to go find the shelf that contains the book themselves. This paper's solution, LIBROB, is a robot that instead takes a book title from the user, searches the library's database for the relevant books, then autonomously guides the user directly to the correct shelf.

II. HYPOTHESES

This study puts forward the two following hypotheses, which will also serve as the metric from which LIBROB's performance will be evaluated:

- LIBROB will substantially reduce the time spent by students looking for a book.
- LIBROB will increase students satisfaction when using library services.

The hypotheses will be tested through experiments described in later sections to either reject or accept the hypotheses put forward.

III. RELATED WORK

Robots have already found their application in public libraries, and some significant advances have already been made. One of the most advanced examples is *Aurora*, the flagship library robot of *Senserbot* [3], which was developed based on a previous work by *A*STAR*, in Singapore. It is capable of navigating a library, scanning the books on the shelves, analysing the result of the scan and generating a report based on the latest library database [2]. It is already operating in Singapore and might soon make its first appearance in Japan and China. The scanning and the identification are both based on Radio Frequency Identification (RFID) technology.

Even though many of the principles are relevant to our project (navigation, book identification, interaction/connection with the database), the aim and human interaction differ fundamentally. *Aurora*'s objective is to find misplaced books and send a report to the librarian, without any interaction with the actual users of the library. On the other hand, our study is focused on improving user experience through interactions with the user. Even so, we have decided to maintain an open approach to future implementation in optimisation of the work of the librarians. Furthermore, *Aurora* is incapable of using the stairs or the elevator, which is an important development aspect for LIBROB as it is being designed to operate in a five-story library.

Another relevant development in this field is the Comprehensive Access to Printed Materials (CAPM) [5]. This is an autonomous mobile robotic library system developed to retrieve items from bookshelves and carry them to scanning

stations located in the off-site shelving facility. Again there are similarities that can be used for this project but also crucial differences. The CAPM project does not directly interact with the user and operates in a off-site facility; LIBROB aims to interact directly with the user and will operate in the actual library

At Aberystwyth University, robotics students have designed a robot named *Hugh* [4]. It is currently still at the prototyping stage and its documentation has not been released yet, but the project has been advertised on the university website, and is worth considering due to the proximity to the LIBROB project. *Hugh* has been described as an *"artificially intelligent library catalogue"*, that *"is able to take verbal book requests, work out where the hard copy is and lead students to the relevant bookshelf"* (Saufenberg, 2016). It has a similar way of recognising books as *Aurora* (it uses RFID); but in contrast it aims at interacting fluently with the users, helping them locate the books in the library, similarly to what our team is trying to achieve. Indeed, the interaction is also based on speech recognition and natural language processing to offer the most comfortable experience possible.

Regarding the technology that we will employ, numerous examples of navigation have been developed for autonomous robots [6]. They rely on sensors, such as laser scanners, GPS and cameras, as well as on a script language that controls route sequencing, obstacle avoidance and junction detection. Similar ultrasonic sensors to the one mounted on the robot that we will employ have been used for robot navigation and obstacle avoidance, and different techniques have been adopted.

IV. HARDWARE DESIGN

The PeopleBot was chosen to be the base hardware platform for our robot. The PeopleBot not only has the appropriate dimensions (see Figure 1) to move around freely through the narrow space between shelves in the library, but also has two powered wheels that allow for it to take sharp turns at reasonable speeds, providing more manoeuvrability than a four wheeled robot, such as the P3-AT.

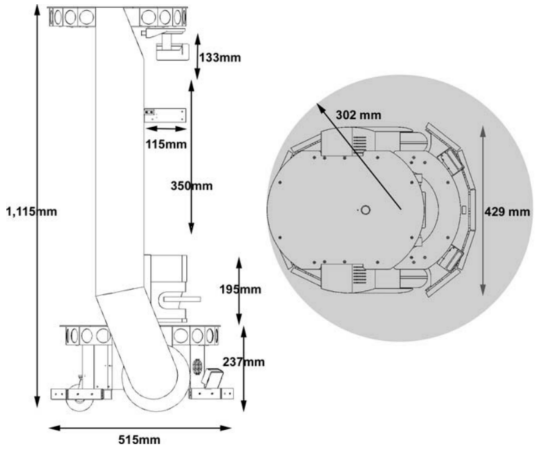

Fig. 1. PeopleBot Dimensions.

In order to enhance LIBROB, multiple hardware components have been added to the PeopleBot. These additions allow the robot to move more freely and make it operate as a standalone device without having to attach a laptop on top of it, thereby making it considerably more user friendly.

- In order to interconnect the electronic devices used to operate the robot, a router was installed onto the moving base. This router connects to the available wireless network of the library in order to give LIBROB internet access but also provides a local network to allow all the system devices to communicate in a safe and stable manner.
- An iPad tablet sits on top of the robot and plays the role of graphical user interface as well as representing the face/head of LIBROB.
- A Raspberry Pi 3b + acts as the base computer for LIBROB. It runs all ROS nodes except for the Navigation node that is taken care of on a remote laptop running on the same local network due to memory performance of the Raspberry Pi.
- To run these scripts, the Raspberry Pi needs a Lidar sensor that scans the robot's surroundings.
- Finally, a microphone coupled with an external sound card is needed for the Speech node since the Raspberry Pi does not include built-in audio recording devices.

V. SOFTWARE SYSTEM DESIGN

The high-level design of LIBROB's software system is depicted in Figure 2. The whole system operates around a central ROS node called "Behaviour". All other nodes in the system primarily communicate with it, since it is the node in charge of taking the major decisions in the process, as will later be described in section V-F.

The next sections of the paper examine in detail the individual nodes that form the high-level design of LIBROB, namely, **Visual UI**, **Speech**, **Database Adapter**, **Locator**, **Navigation**, and finally **Behaviour**.

A. User Interface

The Graphical User Interface (GUI) was implemented as a web application using HTML, CSS, and JavaScript. Most of the design is created with the help of the well known Bootstrap library and a few of the functional processes constitute jQuery code as well. More importantly, the Rosbridge library, which provides a JSON API to ROS functionality for non-ROS programs, is used to interface our front-end application with other ROS nodes through a WebSocket server. The choice of having a web-oriented application was motivated by the idea of having a cross-platform application that would run on many different devices.

The fundamental purpose of the Visual UI node is to make LIBROB more approachable, and allow the user to interact with the robot in an intuitive way, whilst communicating with them through visual feedback and speech synthesis [7]. The

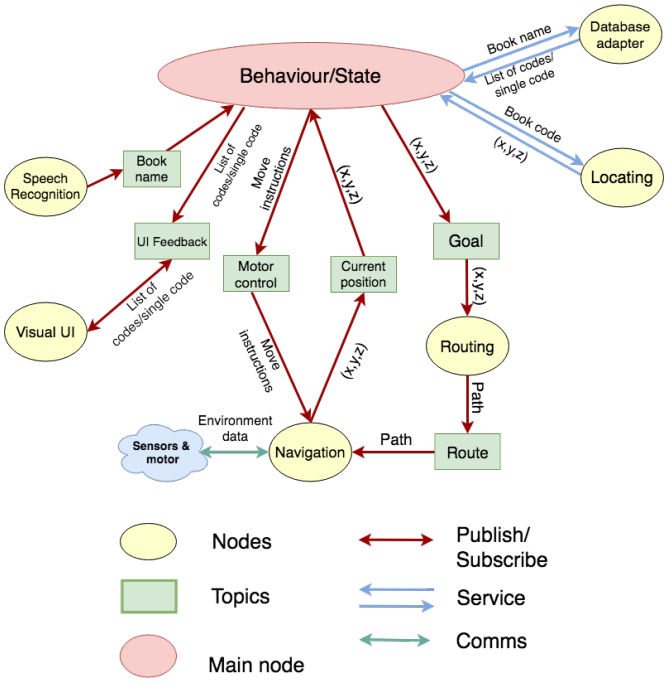

Fig. 2. Software Framework Diagram.

steps of a typical interaction between LIBROB and a user are described in the following paragraphs and figures.

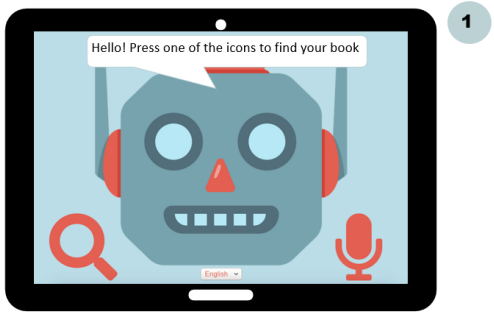

Fig. 3. Initial interface.

1. The user approaches LIBROB and selects one of the two available options: they can either press the microphone icon and ask for a book by talking to the robot, or press the search icon to type in the book title. A drop-down menu is also available at the bottom of the page to allow for different language options. Indeed, LIBROB is able to communicate in different languages, as described later in section V-B. The languages currently supported are: English, French, and Italian.

2. Once LIBROB detects the book title, a list of the books returned by the database is displayed in a table (Bootstrap modal). Relevant information about the title, author, floor number, availability and identification code of each book are all provided in the table. The user is able to scroll through

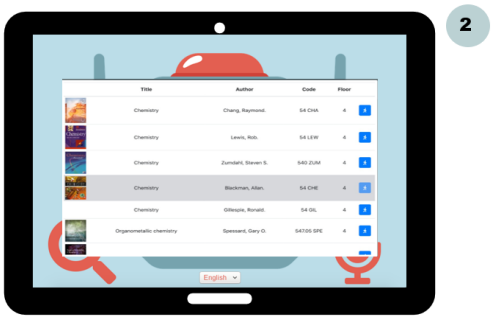

Fig. 4. Result table interface.

the list, find and select the desired book by pressing on the "man" icon next to it.

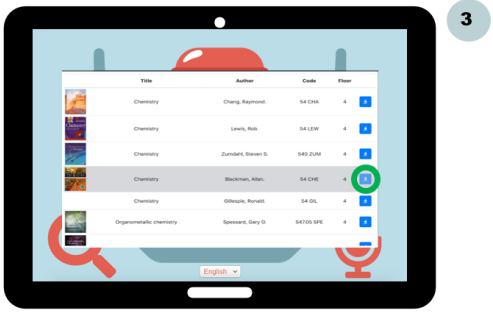

Fig. 5. Result table interface.

3. After pressing the "man" icon, LIBROB will start moving towards the destination shelf where the book is located. At this point, LIBROB provides an audio-visual message saying *lets go!* to tell the user to follow the robot.

Fig. 6. Arrived at destination interface.

4. Once LIBROB reaches its destination, it will announce to the user *We are arrived at your shelf!* and the operation is terminated.

It is worth noting that these specific messages only occur for this specific scenario[1]. However, other types of messages, controlled by the Behaviour node, have been designed to handle different situations, such as when the robot has

[1]Standard scenario with no issues.

not found a book, or when the user is trying to interact with it while it is moving. Moreover, as stated previously, all these messages exist in all of the supported languages (refer to Appendix A), meaning that speech synthesis was implemented in different languages as well.

B. Speech Processing

With the aim of ensuring the smoothest human-centered interaction, much effort was allocated to the Speech module. In order to allow LIBROB to communicate with humans naturally, the team made use of a combination of three technologies within the field of Speech Processing, namely real-time speech recognition, language translation, and natural language processing.

In the process of understanding humans, the Speech node's initial goal is to transcribe[2] what the user says through speech recognition, and publish the book request on a ROS topic that would be fed to the Database Adapter node by the Behaviour node. This task was made robust by ensuring it would operate well even in noisy environments and in environments where noise levels may vary [8]. Initially, this implementation alone formed the entirety of the Speech node. However, this design assumes that the user can only ask for a book by saying the book title and hence does not allow for a natural conversation between the human and the robot.

Therefore, to improve LIBROB's communication capabilities, a natural language processing layer was added to the design, with the use of the Snips NLU PYTHON package [10]. It is based on a machine learning model trained with many examples of how humans could ask for a book orally. The objective of this layer is to extract semantic information on the presence of a book title or an author's name in the transcribed string, in which case the relevant information is sent as a request. Following the implementation of this natural language processing layer, LIBROB was able to understand sentences such as: "*Could you help me find a book on Organic Chemistry written by Alan Turner please?*" or "*I really need a book to study for my Digital Signal Processing exam tomorrow!*".

Moreover, the team decided to add real-time language translation to the Speech node to allow users to talk to LIBROB in their native language. Thus, depending on whether or not the user requests for a book in English, a language translation to English is performed in the background using the speech translation package from [9] and the translated string is fed to the natural language processing layer[3]. A full work flow of the speech node can be found in Figure 7[4].

Finally, the Speech node is also designed to handle the case where the user has to take the lift to get to the right floor. In this scenario, a background listener is triggered as soon as LIBROB enters the lift to listen continuously until it hears the floor number corresponding to the book floor.

[2]Speech-to-text conversion.

[3]The same "English-trained" model is used for any language as it only has to operate with the output of the translation layer.

[4]The example input is a French sentence translating to: "Hello, I am looking for a Mathematics book written by Thomas Murphy!"

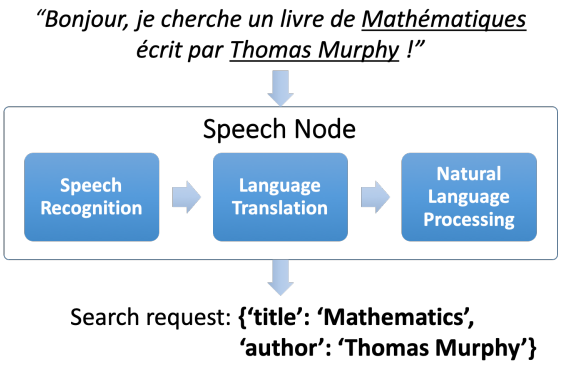

Fig. 7. Work flow of the Speech node.

C. Database Adapter

Determining exactly which book the user is looking for, based on the limited amount of information input during the search process, is crucial to the correct operation of LIBROB. The logical solution to achieve this is to have the robot communicate with a database where all the relevant information of every book in the library is stored.

Due to the large amount of books generally available in a library, storing such a database locally on LIBROB is impractical as it would require a tremendous amount of storage. The method that is currently used in the system is to run an HTTP request script to the library's database Application Programming Interface (API) which will only return information based on the search performed by the user.

The Database Adapter node script can perform the search with title information, author information or both. By adding arguments to the search, it is possible to perform a more specific search. The most important argument added is to limit the search to printed books so that it excludes any journals, articles or e-books. Furthermore, additions or subtractions to the library are recorded and the availability of books is tracked by the library and visible on the API's response. This information can therefore be exploited by LIBROB to enhance the system. The data returned by the search request is then filtered in order to extract the relevant data used to fill up a JSON object with the following keys (see Figure 8):

- title
- author
- code
- floor
- availability
- thumbnail

Once the object is constructed its format can be adapted in order to allow different nodes to use its contents.

```
{
  "title": "Harry Potter and the Order of the Phoenix",
  "author": "J.K. Rowling",
  "code": "800.ROW",
  "floor": "1",
  "availability": "True",
  "thumbnail": "https://proxy-eu.hosted.exlibrisgroup.comexl_
}
```


Fig. 8. Example of one search response returned by the Database Adapter node for a search request with title key "Harry Potter".

D. Locator

The purpose of the Locator node is to provide the bridge between the Database Adapter and the Navigation node. When the Behaviour node requests the service by passing on the classification code of the queried book from the Database Adapter (refer to Figure 5), the Locator node searches through the list of shelves and responds with the corresponding coordinates, to be set as the goal for the Navigation node.

This is made possible by storing the classification codes for the first books of every shelves and the corresponding location coordinates in a YAML file, and converting the set of alphanumerical classification codes into purely numerical codes. Using the new numerical codes, the locator node script can compare their magnitude values to establish an order and identify which shelf the queried book belongs to.

The initial attempt at building the required algorithm involved using regex techniques and libraries to parse the book codes character by character to establish order between the shelves and queried books. This however, proved to be unnecessarily complex compared to the previously proposed method of converting the alphanumerical codes into numbers.

E. Navigation

To be able to move across the library, LIBROB is provided a navigation module based on existing maps of different library floors. Imperial College's central library has been mapped using a 360°Lidar, Synchronous Localisation and a Mapping software (Hector SLAM). Maps are manually corrected and normalised such that lifts'are overlayed on different floor maps.

The robot is localised on a provided map (see Figure 9) using the Monte Carlo method (AMCL). This relies on a clear point cloud provided by the Lidar and reliable odometry provided by the moving base. When the Behaviour module sends a new goal target, the shortest path is computed using the Dijkstra algorithm. Costs used in Dijkstra are derived from two costmaps : a global one and a local one. The global costmap is static and based on the floor map of obstacles inflated. The local costmap is a 5 meter square centered around the robot and constantly updated with current obstacles inflated. When eventually found, the local path is derived into velocity command by the path planner. The navigation module also uses the ActionLib library to provide feedback on the current goal. When the goal is reached, an event is triggered[5].

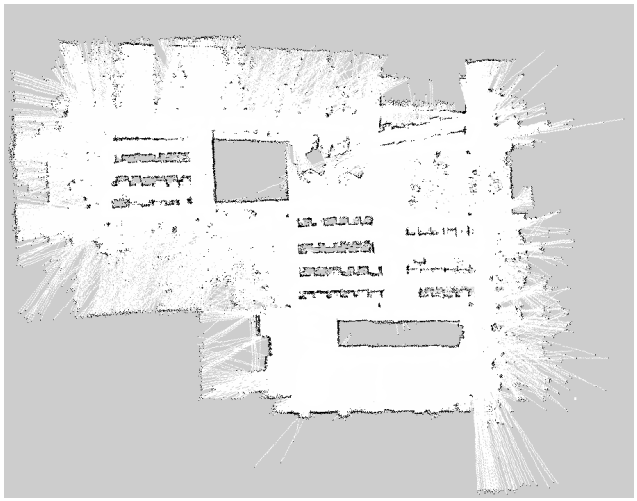

Fig. 9. Map of Imperial College Library's 4^{th} Floor.

F. Behaviour

The behaviour node is responsible for controlling the actions undertaken by LIBROB. Figure 10 shows the design of the state machine implemented in the behaviour node that determines the actions to be taken by LIBROB based on received inputs and measurable goals. The implemented states are as following:

- Idle : LIBROB is waiting for a user to initiate interaction and request a book title
- Display list of Books : The database request results are displayed to the user
- Navigate to Book : LIBROB uses the assigned shelf coordinates for the user-selected book as a navigation goal
- Indicate book : LIBROB notifies the user that the book is located immediate shelf
- Return to station : LIBROB uses the station coordinates to navigate back to it's original station

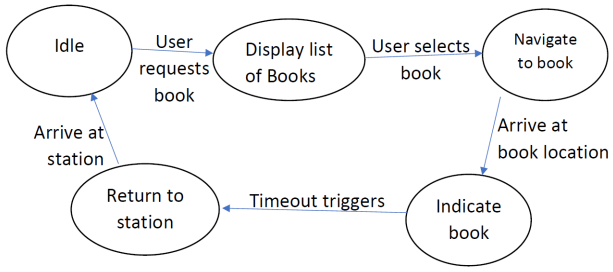

Fig. 10. LIBROB state machine.

[5]This is useful, for example, when we wish to notify the user that the robot has reached the book shelf.

To implement the steps taken to navigate to a book on a different floor, the navigation state is separated into multiple sub-states, as shown in Figure 11, and include the following:

- Move to Lift: LIBROB navigates to the front of the lift doors
- Wait for Lift: LIBROB searches for indication that the lift doors have opened
- Enter Lift: LIBROB enters the lift
- Move to book: the requested shelf location becomes the new navigation goal

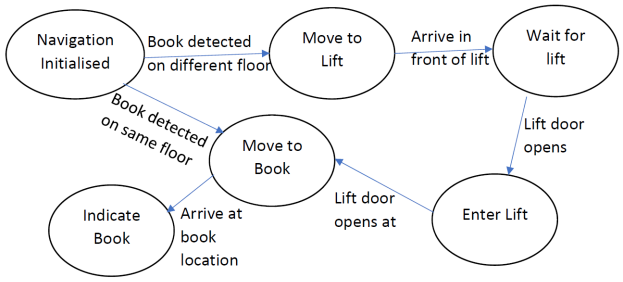

Fig. 11. Navigation sub-states.

VI. RESULTS AND VALIDATION

A. Aim

The aim of the experiment is to test the hypotheses, that LIBROB will reduce the average time students spend looking for a book and increase student satisfaction with the library system. The experiment also aims to learn if users are more likely to use voice interaction when made available, or if text typing is the preferred medium of communication.

B. Setup

The experiments will be conducted on the 4th floor of the Imperial College Central Library to achieve the most accurate and representative data. LIBROB will wait at the entrance to the 4th floor of the Imperial College Central Library, and 40 voluntary participants will be used. Students will be split into two groups and tasked with finding a book of a specific title. The first group, the control group, will be allowed to use the existing library system, but will not be given access to LIBROB. The second group will be given full access to LIBROB's functionality.

C. Data collection

The following data will be recorded during each trial:

1) Average time to find the book of given title.
2) Successful vs failed attempts of LIBROB to guide a user to requested book.
3) Speech vs text typing usage of users with LIBROB.
4) Number of times the user required intervention to successfully use LIBROB.

Furthermore, at the end of each trial, participants in the second group will be given a survey to assess whether LIBROB achieves the required criteria (see Table I for the survey questions).

Survey Question	Answer Format
How satisfied are you with your library experience?	Rating Scale
How useful do you think LIBROB is?	Rating Scale
If you voted for not useful, why do you think so?	Multiple Choice
How likely are you to use LIBROB again?	Rating Scale
How intuitive to use was LIBROB?	Rating Scale

TABLE I
SURVEY QUESTIONS

D. Results

In both groups, every participant managed to find the book they were asked to find. However, the ones using LIBROB spent 47.4% less time in doing so (see Figure 12). The time to find a book using the robot averaged at 92 seconds (standard deviation 24s), while the one recorded without the robot was 175s (standard deviation 40s).

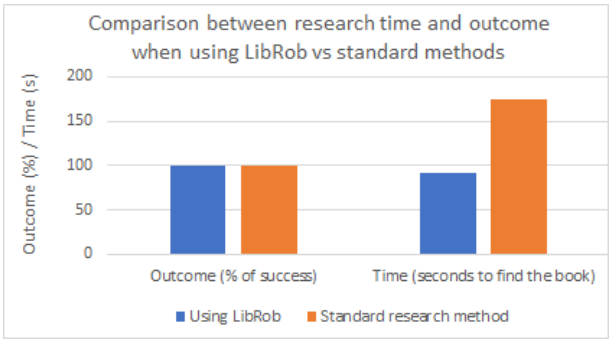

Fig. 12. Success rate and time comparison of LIBROB and the standard search method.

85% of the people who used the current method of research stated that they would have liked to have a better system to find the books in the library, as illustrated in Figure 13.

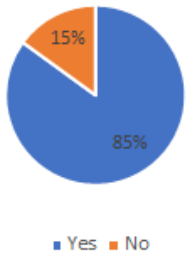

Fig. 13. Answers from the control group when asked if they would have liked to see improvements on the current book location system in the library.

The survey results, as shown in Figure 14, show that 80% of users were satisfied with their experience, 70% found the robot useful, 85% said they were willing to use the robot again and 90% found the robot intuitive to use. Additionally, participants on average required no interventions to successfully use LIBROB, further supporting that it is intuitive to use.

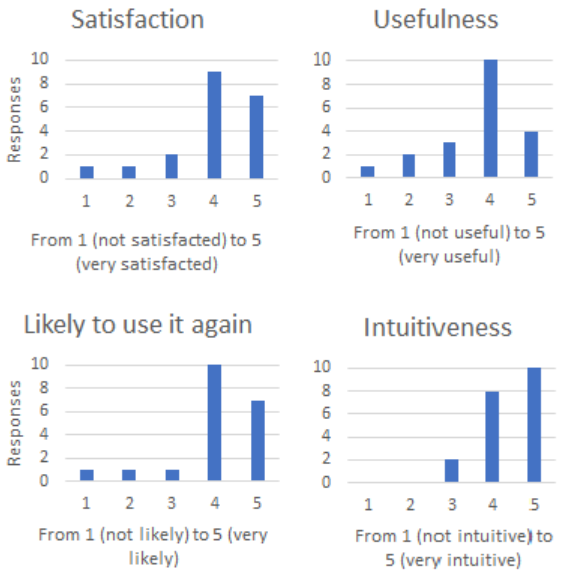

Fig. 14. Responses to the survey presented to the people who used LIBROB.

When comparing the use of text-typing and speech-to-text, only 20% of users used text-typing to search for a book title, suggesting that speech-to-text is the preferred form of interaction with LIBROB. For those who stated they were not satisfied with the LIBROB, the reasons for their answers are shown in Figure 15.

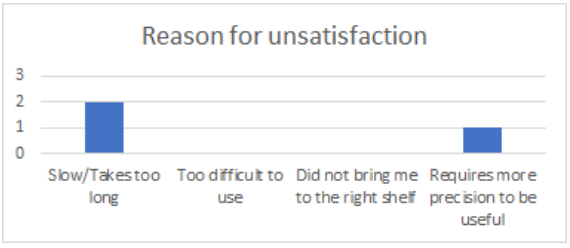

Fig. 15. Reasons for unsatisfaction.

VII. DISCUSSION

The rate of success and the times recorded confirmed the hypothesis that the robot can help users to find the requested books more quickly with the same reliability. In addition, it should be noted that LIBROB also reduces the variance of the search time for a given book. This is expected as once the user inputs their query, the robot guides them to the right shelf, and the variance in time is only introduced when users need to locate the book on the shelf themselves. LIBROB proved to almost halve the time necessary to find a book in the library.

From the survey presented to the control group it is possible to assess that there is a considerable need for a better solution to locate the books in the library, with 85% of the people confirming that an improved system was needed

The responses to the survey given to the users of LIBROB, reveal an overall satisfaction of 80%, with 70% of the people finding it useful. This means that not only can LIBROB pragmatically reduce the time users spend looking for a book, but even in cases where the time spent looking for a book is not significantly reduced, it still ensures a positive experience, as evidenced by the high satisfaction rate.

To understand the reasons behind negative responses, and to better direct future improvement work on the robot, we collected information about the reasons for which users were left unsatisfied. Two of the participants claimed that the robot was too slow or took too long, and one was not satisfied with the precision of the indications received. None of them found issues about the interaction with the robot or reported the robot bringing them to the wrong shelf. The cause for the robot being too slow is primarily due to the nature of the robot base itself. The robot we used, PeopleBot, has a maximum forward speed of 0.8 m/s, which could appear too slow for some users if they are covering considerable distances. On the other hand, the precision issues are largely due to the restrictions of the hardware. Some users found that identifying the right shelf was simply not sufficient, and they wanted LIBROB to point precisely at the right book. This is being considered for future improvements on LIBROB.

The robot proved to be very intuitive to use. Only 10% of the participants were not immediately sure on how to operate it, even though all of them approached it for the first time during the test. 85% of the group said that they would use the robot again, which is an incredibly encouraging result: the robot is not meant to entirely substitute the present book location system, but to provide an improved alternative for the users. Overall, the interactions between the users and LIBROB proved to optimise the book location time and provide a satisfactory experience for the library users.

VIII. FUTURE WORK/IMPROVEMENT

The current development of LIBROB allows the users to be successfully guided towards the book they are looking for. Furthermore, it statistically satisfies the two hypotheses put forward by our study. Nevertheless, additional design-related implementations are still possible, some of which are already implemented as separate functions and thus ready to be integrated in our design.

Firstly, a face recognition algorithm was developed to detect faces of the users through the device's webcam. It would track their position and allow LIBROB's eyes to follow them. This would capture the users'attention and make them feel even more engaged when interacting with LIBROB. Although the code was fully functional on its own, integrating it with

the rest of the system ended up being tricky (see Appendix B) . Therefore, further work will be spent on its integration in order to improve the user-robot interaction.

Secondly, one element that was missing in our tested prototype was the ability of LIBROB to take the elevator. Being able to take the elevator would have allowed a more complete performance of LIBROB since it currently only has access to its initial starting floor. The current implementation would require 5 copies of LIBROB to cover the entire library[6]. All the code concerning the elevator has already been developed (see Appendix C). In fact, LIBROB is able to understand when it has arrived to the correct floor (for further explanation, please refer to section V-B on the Speech node) as well as entering the elevator and updating the floor map once reaching a new floor. The reason why it was not implemented is due to the fact that the lack of an internet connection in the elevator was not taken into account, which is fundamental for the speech recognition package to operate. `Pocketsphinx`, an offline Speech Recognition package for PYTHON, was also tested, however the accuracy of the speech recognition suffered greatly. In the future, a substantial amount of work needs to be spent on elevator handling to fix the present issues and to integrate it in the design.

Finally, a number of trial users suggested the inclusion of a tool to point to the queried book in order to further reduce the time spent looking for it once the shelf has been reached. In order to do so, computer vision could be adopted to recognise the book codes on the shelf and with the aid of an orientation-controlled laser, the robot could be able to point at the right book.

IX. CONCLUSION

In conclusion, LIBROB has been successfully implemented and tested. From the statistical data collected during testing, we have verified and can accept our hypotheses. The time taken for students to locate a book was almost halved when they were using LIBROB; this is a substantial reduction as our first hypothesis predicted. Furthermore, the users were overall more satisfied with their library experience.

Further conclusions can be drawn from the data collected than just proving our hypotheses which prove useful for future researchers in the field. Feedback indicated that many users were happy with the user interface, found it intuitive and felt it enhanced their experience with the library system. While we acknowledge that testing participants were all Imperial College students and not representative of the general public, it is still sufficient evidence that the proposed solution is an efficient solution for this application in the Imperial College Central Library. Finally, constructive criticism was received for possible future improvements of the robot; LIBROB was perceived as being too slow and there was also demand for it to indicate the precise book location instead of just the shelf. The final design of LIBROB is depicted in Figure 16.

[6]One for each floor.

Fig. 16. Final design.

REFERENCES

[1] Di Veroli, C. (2018). Imperial Library. [online] Surveymonkey.co.uk. Available at: `https://www.surveymonkey.co.uk/r/3WVNKCW` [Accessed 31 Oct. 2018].

[2] Li, R. and Ho, C. (2018). AuRoSS: An Autonomous Robotic Shelf Scanning system. IEEE.

[3] Senserbot.com. (2018). SENSERBOT. [online] Available at: `http://www.senserbot.com/#!/our-robots` [Accessed 26 Oct. 2018].

[4] Aber.ac.uk. (2016). Meet Hugh, the robot librarian - Aberystwyth University. [online] Available at: `https://www.aber.ac.uk/en/news/archive/2016/02/title-181095-en.html` [Accessed 27 Oct. 2018].

[5] Suthakorn J. and Chirikjian G. (2002) A Robotic Library System for an Off-Site Shelving Facility, Department of Mechanical Engineering, The Johns Hopkins University, Baltimore.

[6] Aggarwal S. and Priyadarshini M. (2016) Robot Navigation: Review of Techniques and Research Challenges. IEEE

[7] ResponsiveVoice.JS. (2018). ResponsiveVoice Text To Speech - ResponsiveVoice.JS. [online] Available at: `https://responsivevoice.org/` [Accessed 4 Dec. 2018].

[8] Python, R. (2018). The Ultimate Guide To Speech Recognition With Python Real Python. [online] Realpython.com. Available at: `https://realpython.com/python-speech-recognition/#picking-a-python-speech-recognition-package` [Accessed 15 Nov. 2018].

[9] PyPI. (2018). translation. [online] Available at: `https://pypi.org/project/translation/` [Accessed 5 Dec. 2018].

[10] Snips-nlu.readthedocs.io. (2018). Snips Natural Language Understanding Snips NLU 0.18.0 documentation. [online] Available at: `https://snips-nlu.readthedocs.io/en/latest/index.html` [Accessed 3 Dec. 2018].

[11] Adept Mobilerobots for PeopleBot. 2011. Mobilerobots. PeopleBot datasheet `https://www.generationrobots.com/media/PeopleBot-PPLB-RevA.pdf`

APPENDIX A

```
const COMMUNICATION_MESSAGES = {
    'en-US':
    {
        "SEARCHING": "Hmm... let me think",
        "DB_ERROR": "Sorry, I cannot request the database",
        "LOCATOR_ERROR": "Sorry, I cannot find the location of this book",
        "NOT_UNDERSTOOD": "Sorry, I did not understand what you said",
        "FOLLOW_ME": "Let's go ! Please follow me",
        "BUSY": "Sorry, I'm still moving and cannot process your request",
        "HOW_TO_TALK": "I'm listening, what can I do for you ?",
        "TIME_OUT": "You've been too long...",
        "ARRIVED": "Here is your book !"
    },
    'fr-FR':
    {
        "SEARCHING": "Hmm... Laissez-moi reflechir",
        "DB_ERROR": "Pardon, je n'arrive pas    acceder la base de donnees",
        "LOCATOR_ERROR": "Pardon, je ne trouve pas la location de votre livre",
        "NOT_UNDERSTOOD": "Pardon, je ne vous ai pas compris",
        "FOLLOW_ME": "Allons-y! Veuillez me suivre s'il vous plait",
        "BUSY": "Pardon, je suis en mouvement et ne peux pas traiter votre demande",
        "HOW_TO_TALK": "Je vous  coute , que puis-je faire pour vous ?",
        "TIME_OUT": "Vous avez pris trop de temps...",
        "ARRIVED": "Voici votre livre !"
    },
    'it-IT':
    {
        "SEARCHING": "Hmm ..fammi pensare",
        "DB_ERROR": "Scusa, non riesco ad accedere al database",
        "LOCATOR_ERROR": "andiamo! Seguimi",
        "NOT_UNDERSTOOD": "Scusa, non ho capito  c i  che hai detto",
        "FOLLOW_ME": "Andiamo! Seguimi",
        "BUSY": "Scusa sono impegnato non posso processare la tua richiesta ora",
        "HOW_TO_TALK": "Sono tutto orecchie. Cosa posso fare per te ?",
        "TIME_OUT": "Ci hai messo troppo tempo..."
    }
};
```

Fig. 17. Communication Messages in the visual_ui node for English, French & Italian.

```
if (language === 'en-US') {
    responsiveVoice.speak(msg, "UK English Male",  {pitch:1},{volume: 1},{rate: 10});
}
else if (language === 'fr-FR') {
    responsiveVoice.speak(msg, "French Female",  {pitch:1},{volume: 1},{rate: 10});
}
else if (language === 'it-IT') {
    responsiveVoice.speak(msg, "Italian Male",  {pitch:1},{volume: 1},{rate: 10});
}
```

Fig. 18. Speech Synthesis.

APPENDIX B

```
window.onload = function() {
var video = document.getElementById('video');
var canvas = document.getElementById('canvas');
var context = canvas.getContext('2d');
var eye = $(".pupil");
var tracker = new tracking.ObjectTracker('face');
tracker.setInitialScale(4);
tracker.setStepSize(2);
tracker.setEdgesDensity(0.1);
tracking.track('#video', tracker, { camera: true });
tracker.on('track', function(event) {
    context.clearRect(0, 0, 360, 270);
    event.data.forEach(function(rect) {
    context.strokeStyle = '#a64ceb';
    context.strokeRect(rect.x, rect.y, rect.width, rect.height);
    context.font = '11px Helvetica';
    context.fillStyle = "#fff";
    context.fillText('x: ' + rect.x + 'px', rect.x + rect.width + 5, rect.y + 11);
    context.fillText('y: ' + rect.y + 'px', rect.x + rect.width + 5, rect.y + 22);
    var trans_x = (rect.x)/((360-rect.width)/40) - 20;
    var trans_y = (rect.y)/((270-rect.height)/40) - 20;
    eye.css({
        '-webkit-transform': 'translate(' + -trans_x + 'px, ' + trans_y + 'px)',
        '-moz-transform': 'translate(' + -trans_x + 'px, ' + trans_y + 'px)',
        '-ms-transform': 'translate(' + -trans_x + 'px, ' + trans_y + 'px)',
        'transform': 'translate(' + -trans_x + 'px, ' + trans_y + 'px)'
    });
});
```

Fig. 19. Javascript Implementation of the eye-tracking process (commit ID: 7098bbc0 on the visual_ui branch).

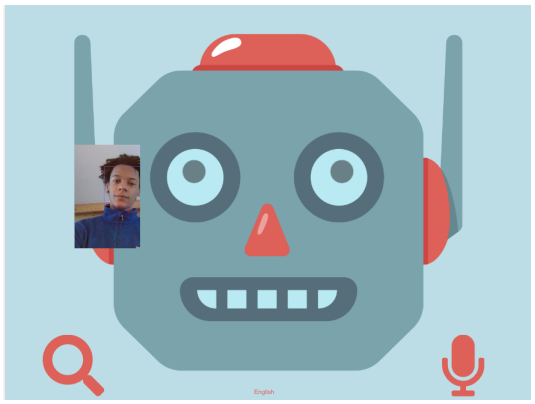

Fig. 20. Eye tracking using Face Recognition (tested in Visual UI alone).

APPENDIX C

```
def sub_lift_callback(self, lift_msg):

    if lift_msg.type == lift.LIFT_TRIGGER:
        if json.loads(lift_msg.payload)['state'] == True:
            r = sr.Recognizer()
            self.stop_function = r.listen_in_background(sr.Microphone(), backgroundCallback)
        else:
            self.stop_function()

def backgroundCallback(recognizer, audio):

    try:
        string = recognizer.recognize_google(audio).lower()
        substring= "floor"
        print("Background listening recognised: " + string)

        if substring in string:
            idx = string.find(substring)
            newstring = string[idx:]

            if "1" in newstring:
                self.publish('lift', lift_msg.CURRENT_FLOOR, json.dumps({'floor': 1}))

            elif "2" in newstring:
                self.publish('lift', lift_msg.CURRENT_FLOOR, json.dumps({'floor': 2}))

            elif "3" in newstring:
                self.publish('lift', lift_msg.CURRENT_FLOOR, json.dumps({'floor': 3}))

            elif "4" in newstring:
                self.publish('lift', lift_msg.CURRENT_FLOOR, json.dumps({'floor': 4}))

            elif "5" in newstring:
                self.publish('lift', lift_msg.CURRENT_FLOOR, json.dumps({'floor': 5}))

    except LookupError:
        print("Oops! Didn't catch that")
```

Fig. 21. Elevator handling through Speech Recognition (commit ID: 7d01c921 on the speech branch).