AutoBookFinder: A Case Study of Automated Book Rack Identification in Library through UGV

Udit Nath Bhaskar Jyoti Medhi, Anuradha Deori, Maharaj Brahma, Mwnthai Narzary,, and Pranav Kumar Singh

Central Institute of Technology Kokrajhar, BTR, Assam-783370, India

Abstract. The misplacement of books in libraries often results in wasted time when searching for a specific book on the wrong shelf. To address this issue, we created a novel, low-cost prototype Unmanned Ground Vehicle (UGV) equipped with a camera and Wi-Fi modules. The UGV wirelessly sends data to a local server, which uses image processing to determine the location of the book. A web user interface was created to enable users to easily locate the exact shelf of the desired book. This solution has not been previously explored and was tested in our Institute's library. The adoption of UGVs greatly improved the overall user experience, transforming the manual and time-consuming process of book-finding into an automated and time-saving operation.

Keywords: Smart Library · UGV · Image Processing · Flask · EasyOCR · Arduino UNO

1 Introduction

The Central Library of Central Institute of Technology Kokrajhar (CITK), Assam, India has a collection of approximately 0.3 million books [1]. The library organizes each book into proper places based on its genre and category, and book issuing and returning activities are digitized. Users are required to use their library cards to perform any activity in the library, and the cards are provided by the library. The library patrons handle the entire process of issuing and returning books, while the library staff supervises. On an average day, the library receives approximately 300 visitors and processes about 150 issue requests. The library is divided into different shelves according to the department to which the book belongs, and each shelf has different racks where the library staff stores the books after they are returned by users. However, the problem arises when users search for a particular book. The books are randomly placed on a rack when returned, making it difficult for library users to find the required book in its place. Moreover, the library racks do not have a specific naming system, so library patrons spend a long time searching for books. It has also been found that sometimes users cannot locate a required book even when it is shown as available in the library database.

The main objective of this project prototype is to create a system that informs library patrons of the precise location of a book by providing its rack number. This will be accomplished by using an Unmanned Ground Vehicle (UGV) [4] to inspect the library racks regularly and capture still images of them. The system will then process the images to extract book details and rack numbers, which will be fed into the database. The entire system will operate wirelessly, without requiring an internet connection. A user-friendly web interface has been developed to simplify the process of searching for a book name in the database, which will direct patrons to the appropriate shelf. This system will also assist the library in reducing its dependence on staff to manage library operations, such as locating and organizing books.

The remainder of the paper is structured as follows. Section 2 discusses related and relevant works. Section 3 outlines the materials and methods employed in this study. Section 4 provides a detailed description of our experimental setup. Section 5 presents our findings and provides a discussion of the results. Finally, in Section 6, we summarize our conclusions and offer suggestions for future research.

2 Related Works

In this section, we review the existing literature on book locating solutions in library racks and shelves. Automatic library book detection systems using book call numbers have shown to be effective, and various research works have been conducted in this area. For instance, Duan et al. [5] proposed a line segment detector with color segmentation and slanted angle removal techniques, which significantly enhanced the performance. The authors presented a modified contours clustering method to extract call numbers completely and achieved an accuracy of 97.70% using the proposed technique, compared to 87.77% accuracy with a combination of Hough and color. The authors also highlighted that smartphones cannot capture all books in a single row shelf.

Yang et al. [9] proposed the use of Convolutional Neural Networks (CNN) and Recurrent Neural Network (RNN) deep learning-based techniques for spine text information extraction. The authors achieved an accuracy of 92% and 91% on Tsai2011 (Text) dataset and their dataset, respectively, using a large physical library database. In another work, Ekram et al. [6] used image segmentation and OCR techniques to perform book indexing and organize mismatched library books. However, the authors noted that the OCR algorithm is unable to detect the book call number of images captured from different angles, achieving a 76.08% accuracy on a test set of 100 image samples. The authors suggested that the use of Hough transformation, lines, and contours for segmentation can lead to better results. Cao et al. [3] introduced BSRBOT, a book spine recognition framework that performs image acquisition, segmentation, call number recognition, and spine image matching from an existing database. The authors achieved an accuracy of more than 90% and an average time per spine of fewer than 300 milliseconds.

Previous work in this direction mainly focuses on using smartphones for image capturing and processing which are costly. They also focus on the use of book call number for book identification. However, in this work, we perform experiments on the physical library of CITK, where the books are not maintained by the call number. Instead, each book has a Radio Frequency Identification (RFID) chip [8] that is issued to issue and return the book through a self-kiosk machine. This is particularly challenging due to the non-existence of rack names and the lack of a mechanism to locate the misplaced book. Hence, in this work, we proposed low-cost solution that leverage the use of UGV to traverse the library area, existing OCR technology, and rack naming. The proposed solution works completely on a local server without the need for cloud processing or the internet. The overall objective of the work is to reduce the time wasted by the students in finding the books. To address this, we also provide a web interface to query the rack number using the book name.

3 Proposed Mechanism

The proposed UGV-based mechanism is depicted in Figure 1. The UGV traverses the path around the bookshelves, taking pictures of the racks. The images are sent to the system wirelessly and stored in a local directory. The main system does routine checks on the database. The system requests images from the local directory. Upon locating the images, the system performs image processing and text extraction on the images. The resultant data is then fed to the local database. The system also provides a User Interface (UI) that allows us to search for a particular book. The system fetches the search result and displays the proper location and rack number of the entered book. The overall working principle of the prototype is divided into six steps, which are as follows:

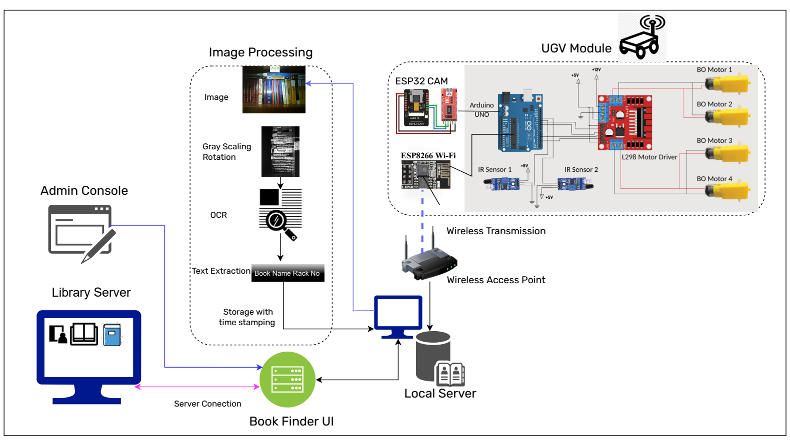


Fig. 1: System Architecture

1. **UGV Traversal and Image Capturing:** The UGV is turned on at a specific time of the day, and it traverses the path, behaving as a line follower. The UGV makes occasional calculated steps in front of the library racks taking images of the books in the racks. The UGV continues traversing and taking pictures of the racks until the entire path is traversed.
2. **Image Transfer and Storage:** The UGV has a Wi-Fi interface to connect to the Library Wi-Fi access point (AP). The images captured by the camera module mounted on UGV in step 1 are transferred to the local server connected with UGV wirelessly (having a fixed IP) via Library AP. The system receives the captured images and stores them in a local directory of system. The local server is designed to check for the images once a day at a specific time. The system checks for the directory wherein the images are stored and fetches them one by one to perform necessary image processing.
3. **Image Processing:** The local server applies the image processing techniques to find the best results and perform image enhancements on stored images captured in different lighting conditions and with varying book orientations.
4. **Text Extraction:** In this step, text extraction is applied to the processed images to get the book name and corresponding rack number in the text format.
5. **Data Cleaning and Storage:** Before feeding the extracted text to the database, it is crucial to ensure that the data is clean, readable, and correct. To serve this purpose, the local server uses a cleaning process to remove unwanted characters and digits and a spellchecker to correct the words. The cleaned and corrected text is stored in a separate database that contains the book name and corresponding rack name. A timestamp is also maintained to keep a record of the entry.
6. **Book Finding through UI:** After coming to the library, the user opens the UI to search for the book. Since the UI is connected to the backend database, it returns the corresponding rack number for the entered book by the user.

4 Experimental Setup Details

This section of the paper provides selected operational area details for our case study, UGV prototype, and software details that were used to conduct the experiment and demonstrate the effectiveness of the proposed solution.

4.1 Operational Area Details

The experiment was conducted around the Computer Science & Engineering (CSE) department shelf in the CITK digital library. It has a total area of $79.58m^2$ where the perimeter of each column is 27.648m.

1. **Shelf Distribution:** Each shelf has a length of 12.192m and a height of 2.13m. Figure 2.a gives a better idea of the shelves.

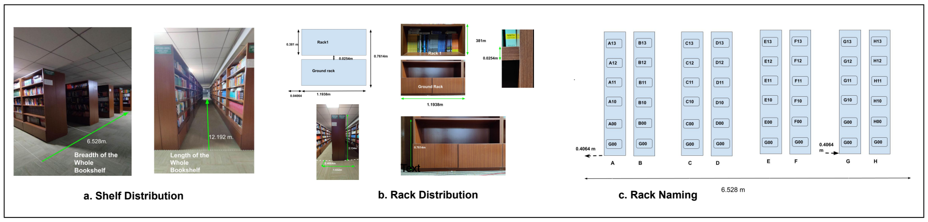


Fig. 2: Operational Area selected for the Experiment

2. **Rack Distribution:** The shelves in the CSE Department had 5X9 (5 rows, 9 columns) rack distribution concerning rows and columns. Each of the racks had a length of 1.91m and a width of 0.381m, as depicted in Figure 2.b
3. **Rack Naming System:** Since, at present the library does not utilize rack naming system hence, it is difficult to organize the book shelves. The proposed rack naming system was derived from the 2-D array-like structure of the racks, as depicted in Figure 2.c Each of the shelves has been given a name A to H, and the format was decided as [shelf name][i][j], where i and j stand for rows and columns, respectively. To place the naming system physically we use sticky notes. We wrote various rack name and number using a black ink on yellow sticky notes and placed in all the shelves inside our operational area. This is done to improve the contrast.

4.2 UGV Prototype

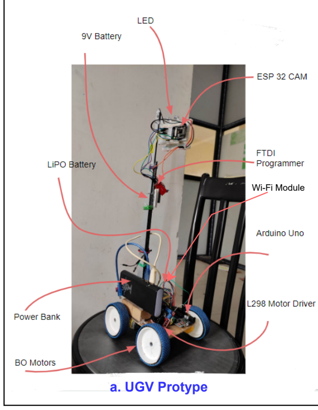
The UGV prototype (Figure 3.a) developed for the experiment is of the autonomous kind. It was controlled via an Arduino UNO, mounted with an ESP32 camera module for capturing the images and transferring them via its Wi-Fi module. The UGV is lightweight and small, with a length of 0.11m and a weight of 1.186g. The camera mount had a Body-and-Arm in RR with an end effector. It is battery-powered and can be plugged into power after its complete traversal. The hardware components used in UGV are listed in Figure 3.b.

4.3 Software Details

We used various software to implement the entire operations, such as for UGV function automation (traversal, stoppage, image capturing, forwarding, etc.), image processing and text extractions, database design and user interface. The details are listed in Table 1.

5 Results and Discussion

The UGV traverses the operational area through a line follower path placed in the operational area as shown in Figure 4.a. It starts capturing images as shown



Sl	Components of UGV	Quantity	Functions
1	Arduino UNO	1	MCU Controls the working of the UGV.
2	ESP32-CAM	1	Camera module for image capturing
3	FTDI Programmer	1	TTL serial communications handling
4	L298 Motor Driver	1	Controls motor speed and direction
5	BO Motors	4	Helps in the locomotion of UGV
6	LiPO Battery	1	Used to power motor driver
7	Power Bank	1	Used to power ESP32 Module
8	9v Battery	1	Used to power LED
9	LED	3	Used for better image quality
10	ESP8266	1	Wi-Fi Module

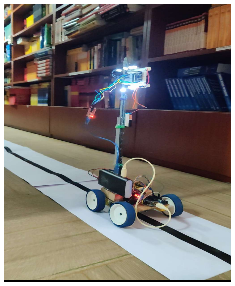
b. Hardware Components used in UGV

Fig. 3: UGV Prototype and its components

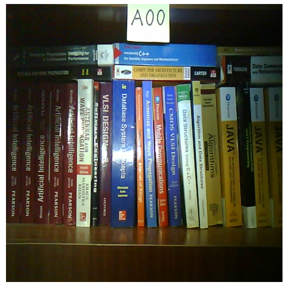
Table 1: Software Details

Sl No.	Software Name	Functions	Device
1	FTDI Programmer	Serial communications handling	UGV
2	Arduino Software (IDE)	Automate the entire UGV Operation	UGV
3	Tesseract	To detect orientation and script of image	Local Server
4	OpenCV [2]	Image Processing	Local Server
5	Easy OCR	Extraction of data	Local Server
6	Auto Correct	For spell checking purpose	Local Server
7	MySql	To design a database to hold the book name and rack name	Local Server
8	Flask [7]	To design the web interface	

in Figure 4.b. The light quality in a closed indoor library environment affects the image quality. Hence, at the local server, we tried two image pre-processing techniques: grayscaling and binarization. The grayscale image is shown in Figure 4.c. We extract the rack name using Tesseract OCR. However, since the books are stacks in vertical fashion. For extracting the corresponding book name, we use Tesseract OCR to detect images containing in a vertically stacked fashion. Next, we check for the orientation of the image if it is vertical and then rotated in a counterclockwise direction. This image is then sent for text extraction. The extraction is performed by EasyOCR. The extracted book name by the OCR suffers from missing characters. To address this, we use a spell checker for OCR post-processing. The *Rack name* and *Book name* information are then stored in the relational database along with the timestamp.



(a) UGV Traversal in Operational Area



(b) Image Capturing

(c) Image Processing

Fig. 4: Results

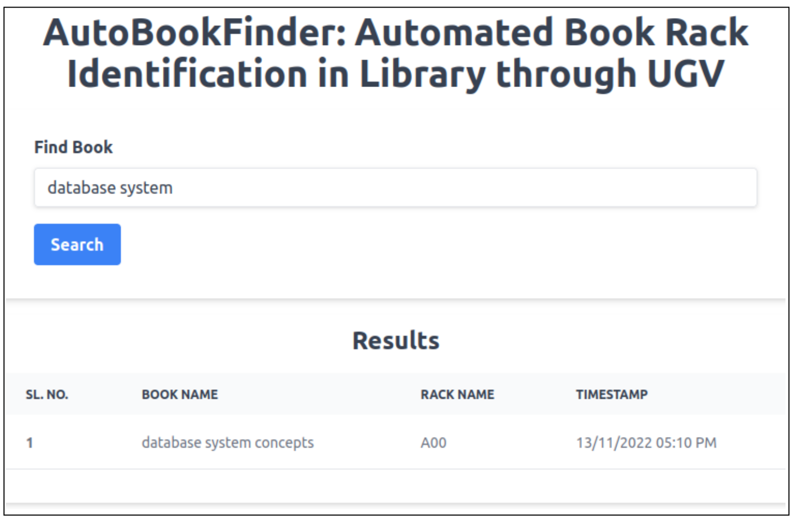


Fig. 5: Web Interface of AutoBookFinder

For accessing the rack information of a particular book, we build a web application that fetches information from the local server. The web application is built using a python based flask framework and contains a search functionality using for book. The resultant rack name is then fetch from the database and shown to the user as depicted in Figure 5. The UGV after traversing the operation area returns back to its starting location. Since physical libraries are very dynamic in nature and there is frequent book issue-return, we setup the UGV to be automatic traverse in the operational area in the time period when the library is less busy.

6 Conclusion

In this work, we present a novel solution for automatic book rack identification through the integration of UGV, image processing technologies, and a rack naming system. The proposed solution is cost-effective and efficient. It requires less human intervention and improves the library user experience.

In this case study, we have explored the bookshelf at the level reachable by the camera. However, the bookshelf can have multiple stacks (top to bottom). Images of such a bookshelf can be captured by an adjustable camera mounted on UGV. Lighting conditions greatly impact the OCR quality, and we address the issue through the use of LED and require prior adjustment based on the rack positions. Therefore, improvements in the camera and LED module can be made to improve the lighting and OCR performance. At present, the proposed system architecture does not consider a docking station for UGV. Hence, a docking system with automatic charging can improve UGV's productivity and lifecycle. In this work, we only extracted the book name along with the rack name. However, it can be extended to capture the associated publisher information. Another challenge that we will consider in our future work is how to deal with the improper orientation of a misplaced book. Furthermore, the web interface currently only allows for searching through book name. It can, however, be extended to support search through title and publisher information. Furthermore, CITK library books are tagged using RFID technology, and UGV mounted RFID reader can be used to read book name as done by Zhang et al. [10]. This approach is scalable when the books are kept in an unordered fashion as well.

References

1. Library, Central Institute of Technology Kokrajhar. http://centrallibrary.cit.ac.in/ (2022), [Accessed 12-July-2022]
2. Bradski, G.: The opencv library. Dr. Dobb's Journal: Software Tools for the Professional Programmer **25**(11), 120–123 (2000)
3. Cao, L., Liu, M., Dong, Z., Yang, H.: Book spine recognition based on opencv and tesseract. In: 2019 11th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC). vol. 1, pp. 332–336 (2019). https://doi.org/10.1109/IHMSC.2019.00083
4. De Simone, M.C., Guida, D.: Identification and control of a unmanned ground vehicle by using arduino. UPB Sci. Bull. Ser. D **80**, 141–154 (2018)
5. Duan, X., Zhao, Q., Anwar, S.: Identifying books in library using line segment detector and contour clustering. In: 2012 7th International Conference on Computing and Convergence Technology (ICCCT). pp. 998–1003 (2012)
6. Ekram, M.A.U., Chaudhary, A., Yadav, A., Khanal, J., Asian, S.: Book organization checking algorithm using image segmentation and ocr. In: 2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS). pp. 196–199. IEEE (2017)
7. Grinberg, M.: Flask web development: developing web applications with python. " O'Reilly Media, Inc." (2018)

8. Roberts, C.M.: Radio frequency identification (rfid). Computers & security **25**(1), 18–26 (2006)
9. Yang, X., He, D., Huang, W., Ororbia, A., Zhou, Z., Kifer, D., Giles, C.L.: Smart library: Identifying books on library shelves using supervised deep learning for scene text reading. In: 2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL). pp. 1–4. IEEE (2017)
10. Zhang, J., Liu, X., Gu, T., Zhang, B., Liu, D., Liu, Z., Li, K.: An rfid and computer vision fusion system for book inventory using mobile robot. In: IEEE INFOCOM 2022 - IEEE Conference on Computer Communications. pp. 1239–1248 (2022). https://doi.org/10.1109/INFOCOM48880.2022.9796711