An Autonomous Robot with Self Navigation and Book Placement for Library

Abdul Haseeb
Department of Electrical Engineering
Namal University
Mianwali, Pakistan
haseeb2018@namal.edu.pk

Muhammad Ramzan
Department of Electrical Engineering
Namal University
Mianwali, Pakistan
ramzan2018@namal.edu.pk

Hamza Zad Gul
Department of Electrical Engineering
Namal University
Mianwali, Pakistan
hamza.zad@namal.edu.pk

***Abstract*— Placement of books in the library is a tedious and time-consuming activity. It requires human intervention with knowledge of rules of book placement and library database. To save money, time, and effort, we propose to automate this task with an "An Autonomous Robot with Self Navigation & Book Placement for Library. "The autonomous robot will scan the spine label of the book via a camera and information will be extracted using information extraction and book finding algorithms. Information extraction from spine labels is cheaper than placing RFID tags. Once it has the required information, it will navigate to the desired shelf. We have devised a Modified Line Following Algorithm which is easily scalable for other applications. Our robot is convenient and easy to use. It requires minimal human intervention and will save time and automate a repetitive task.**

Keywords—Robotics, Cartesian Robot, Robot Design, Image Processing, OpenCV, Self-Navigation, Pick and Place

I. Introduction

Placement of books in the library is a tedious and time-consuming activity. It requires human intervention with knowledge of rules of book placement and library database. To save money, time, and the effort, we propose to automate this task with an "An Autonomous Robot with Self Navigation & Book Placement for Library."

There are international standard classification systems which decides the placement of book in the library. They help with arranging and retrieving books efficiently. A classification system uses letters and numbers to arrange the books on the same topic are together. All information is on a label, usually on the back of the book. These labels are called spine labels. Below is an example of a sorted array of books according to their spine labels.

Fig. 1. Sorted shelf according to the spine labels

Whoever wants to place the book in the library has to memorize the rules and then roam in the library to find the right location. This research paper aims at providing an autonomous solution to automate task of placing books in the library

A. Problem Statement

Placement of books in the library is arduous task. There are robots that automate these tasks, but they are RFID or barcode scanning based. Which requires a specialized reader, a database, and a hardware setup to decode the information. For these solutions to work, all books must have RFID tags on them, which becomes costly

B. Proposed Solution

Each book already and must have uniquely identifiable spine label. The proposed system uses the spine labels to get the position of the book. For navigation to the right shelf, it uses a modified line following algorithm.

By using spine labels, multiple books can be processed together, hence increases speed in the book search.

To adopt the solution, one will not have to place separate identification tags on all the books of the library.

II. Related Work

First, there were card catalogers to manage the library. Then after 1970s computer technologies started taking part in the library automation. Now, there are library management systems that are capable of checking books in/out, updating and maintaining records and cataloging new entries. *Libman and Koha* are some of the famous Library management systems.

Library management system manages all the software related work and maintains the database as well. But the work that demands more physical effort are yet to be automated. These works include, picking and placing books in the shelves and rearranging incorrectly placed books.

Singapore's Agency for Science, Technology [1], proposed a robot to inform about incorrectly placed books. They have developed a robot, a robot that can wander among the stacks at night, scanning shelves for misplaced books. When the robot finds an incorrectly placed book, it informs the librarian. This librarian does not have to roam in the whole library; instead, he can only go to the pointed shelves.

Mayank Pathak et al [2], have proposed a "Library Management Robot ", their robot scans RFID tags from the book, refers to the database to retrieve the correct location and then navigates using the Line Following algorithm provided that the no book in the library is incorrectly placed. They have used proximity sensor to avoid any obstacle.

Suliman Mohamed Fati et al [3], "Automated Library System Using SMS Based Pick and Place Robot." This robot was line following and had a gripper assisted with a pulley system. The only difference with the previous robots was SMS-based working. Users had to message the book's number to the system, the robot turned on and provided the book to the counter.

Till now, navigation of the library management robot has remained a great concern for the researchers. Md. Rezaur Rashid et al [4], they have attempted to make a self-navigation algorithm. They considered the library as a 2D array of bookshelves. Their robot used 5 ultrasonic sensors to measure the distances of its surrounding objects. These measured distances are prerequisites of achieving all the necessary performances, such as calculating the current row-column position of the robot, helping the robot to move in a straight path, making an accurate turn, etc.

Information from the spine label is extracted via image-to-text algorithms. Nevetha M P [5] have proposed a technique that extracts information from the cover of the book. Because the previous systems were only for the covers that are straight and do not detect that are inclined. In their solution, they have used Line segment detection (LSD) algorithm, Line Segment filtering and then OCR for text extraction.

III. METHODOLOGY

Robot has a camera that takes picture of the spine label, from which information is extracted using information extraction algorithm. We have used OpenCV, tesseract and python to design information extraction algorithm. Extracted information is used to determine the location of the book. Then, robot starts moving and finds the right position by a book searching algorithm. After reaching the right position it places the book with the help of mounted arm. Project's flow diagram is as follows

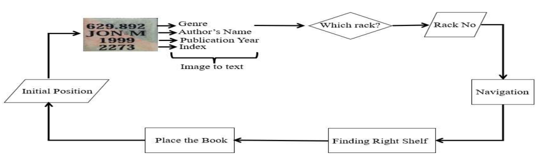

Fig. 2. Project's flow diagram

A. *Robot's Arm*

Robotic arms are used to minimize the human effort and increase the productivity and precision of the operations. We have designed the robotic arm to pick and place the book.

Fig. 3. Robot's Arm

1) *Axis of Motion/Movement*

Axes of motion tell about maneuverability of the robotic arm and refers to the moveable joints of the robotic arm. We have designed an Arm which has 2 axes of motion.

Axis 1 — Moves back and forth (X-axis)

Axis 2 — Raises and lower the gripper (Y-axis)

Can be seen from the figure(no) that it has two segments, one moves in X-axis while other moves in Y-axis. Movement in both axes are controlled by stepper motor and lead screw. Lead Screw transforms rotary motion of the stepper motors into linear motion of the arm. Stepper motor moves the arm to accurate position

2) *Work Envelope of Arm*

Working envelope is the set of points an arm can reach in a space. Working envelope of the designed robotic arm is a square. It can reach all points in a 18x18 inches flat square.

Fig. 4. Work Envelope of the arm

3) *End Effectors*

End effectors are the grippers which are used to hold the objects. We have used two grippers. Two grippers hold the book tightly. Because of using two grippers, book cannot fell in left or right direction and its center of gravity is tapped between two grippers. Grippers are controlled by servo motors.

Fig. 5. Claws opened

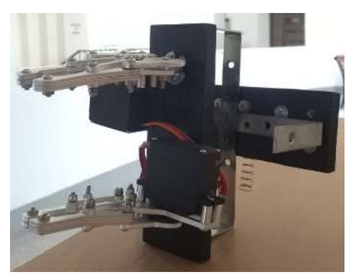

Fig. 6. Claws Closed

B. *Mobile Base*

The robot has a double-decker base. The upper deck is for the arm, while the lower one is used for placing the circuitry and motors. Robotic arm is 18 inches wide, so the width of upper base is 18 inches. Height of the Mobile base is 7 inches from the ground. Motors are attached to the front wheels of the base. DC geared motors are used for higher torque.

C. *Information Extraction from the Spine Label*

Image of the shelve is taken using RaspberryPi Camera. We have devised an Algorithm to extract information from the image. Algorithm has two main parts,

- Book Spine Extraction (Segments the image into rectangular regions of book spines)
- Book Spine Text Recognition (Text extraction from the segmented rectangular regions)

Flow Diagram is as follows

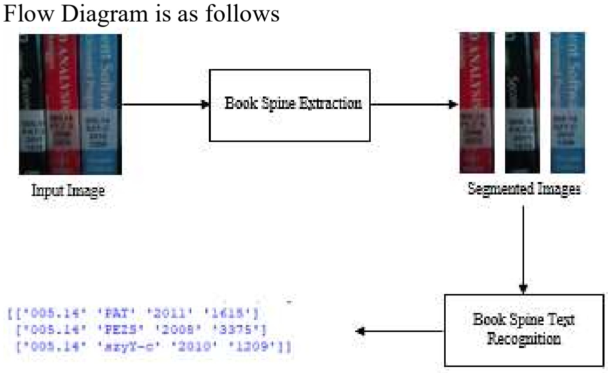

Fig. 7. Block Diagram of Information Extraction

More details about book spine extraction and the book spine text recognition are in the result section.

D. *Navigation Algorithm*

In Robotics, navigation means the robot's ability to plan a path towards some goal location. Many techniques can be used for navigation. We are using line following technique for the navigation because of the following facts.

- Position of racks in the library don't changes by time.
- Moreover, if path of the robot is specified, it makes the process fast, and obstacles can easily be avoided.

We have devised a "Modified Line Following Algorithm", by which robot would not need to roam in the entire library but can easily locate the position of different racks by recognizing the patterns on the path.

IV. Results and Discussion

A. *Information Extraction from the Spine Label*

Information from the spine label is extracted in different stages. We have designed a system, that takes image of the rack as input and returns a list that contains textual information of the spine labels. These stages are discussed below.

Input image is taken via RaspberryPi camera.

1) *Book Spine Extraction*

Image of the shelf contains multiple books, to get better and understandable results, OCR must be given image of a single book. Following procedure is followed to get rectangular segmented images of the separate spine.

a) *Conversion to grayscale*

Image is converted into grayscale, as edge detection algorithm requires it in the greyscale to function properly

Fig. 8. Conversion into the grayscale

b) *Edge Detection*

Edge in the image is a change in the image intensity. Edges in the image are detected using **canny edge detection**. It finds edges based on their derivative and intensity gradients. Noise in the image is removed because canny edge uses derivatives to find intensity gradients so is highly susceptible to the noise [6].

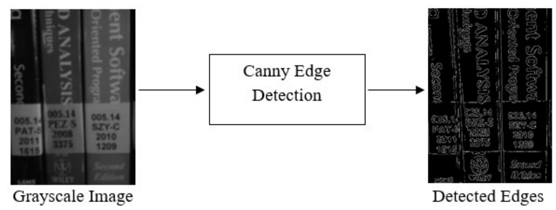

Fig. 9. Edge detection using canny edge

c) *Boundary Detection*

Hough Line transform is used to detect the vertical boundaries of the book. Image with detected edges is given to the Hough Transform, which maps each point in the edge map to a line in parameter space. In OpenCV python, it works on polar coordinate system. Intersection of lines in the parameter space tells whether the points are on the line or not [7]. Function of Hough transform return vectors containing theta and rho. Where, rho is the distance of line from the origin (top left pixel of the image) and theta is the line rotation angle in the radians. Based on the values of rho and theta, lines are drawn over the image.

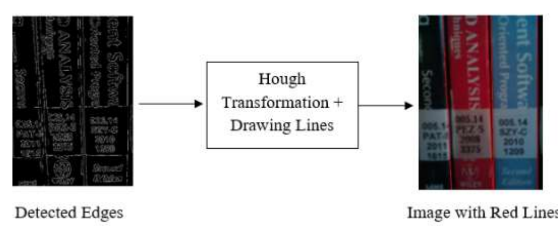

Fig. 10. Boundary detection through Hough Transformation

d) *Segmentation*

Book spines are segmented as the region sandwiched between two lines. We have devised an algorithm which segments each spine based on the assumption that width of each book would at least be 50 pixels. Based on this assumption, sandwiched portion is cropped.

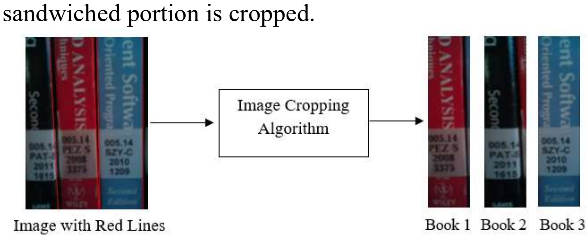

Fig. 11. Cropped Images

2) Book Spine Text Recognition

Optical Character Recognition (OCR) is the technique that is used to extract text from the images [8]. In python it can be done using tesseract. Segmented images are given to the tesseract which returns the text. There are different configuration modes to operate the tesseract, these modes are called page segmentation modes. As the text to extract is a uniform block of text, we have configured it to page segmentation mode 6. Results are as follow

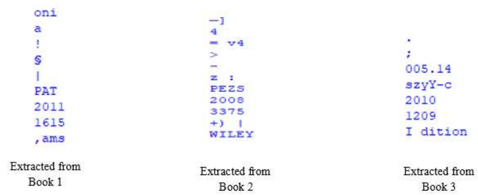

Fig. 12. Text before filtering

It can be seen from the above results, that extracted results contain noise as well, so they need further processing.

a) Text Filtering Algorithm

In further processing, following assumptions have been made to make it noise free

- Spine Label can only contain numeric value and the alphabetic characters
- There would be a hyphen (-) in the author's name
- Genre, Author name, year and index will always be at new line and adjacent to each other

Based on the above assumptions, noise free text is obtained.

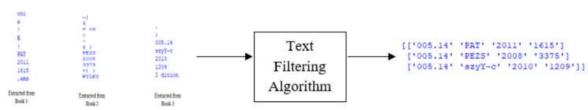

Fig. 13. Filtered text

3) Book Finding Algorithm

This algorithm tells about the position of the book in a shelf. It takes list of textual information of spine labels and information of the book in hand as input and returns the position of the book in hand.

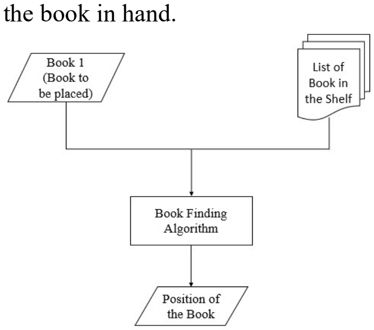

Fig. 14. Boundary detection through Hough Transformation

Book Finding Algorithm looks for the position of algorithm by comparing the sorted array of book.

Variable book contains all the information extracted from spine labels of the book

```
# books = [Genre  ,Author, 'Year' , 'Index']
books = [['005.14', 'PBX', '2011', '123'],
         ['005.14', 'qBX', '2012', '123'],
         ['005.14', 'sBX', '2011', '123'],
         ['005.14', 'uBX', '2011', '123']]
```

a) Case 1

Book in hand has the following information

```
#         [Genre  ,Author, 'Year' , 'Index']
book_in_hand  = ['005.14', 'qBx', '2011', '124']
```

Result

```
Insert before
writer::  qBX
year:: 2012
index 123
```

b) Case 2

Book in hand has the following information

```
#         [Genre  ,Author, 'Year' , 'Index']
book_in_hand  = ['005.14', 'uBx', '2011', '124']
```

Result

```
Insert After
writer::  ubx
year:: 2011
index 123
```

B. Modified Line Following Algorithm

We have modified the line following algorithm. Our algorithm has the following features

- Tells the robot about the position of different racks in the library
- keeps the robot on the black line

This is technique uses IR sensors. We are using three IR sensors. IR sensor gives Boolean output where '1' indicates that the sensor is on the black line and '0' indicates other surfaces. Sensors are at the front of the robot, the position of sensors on the robot is as shown in the figure

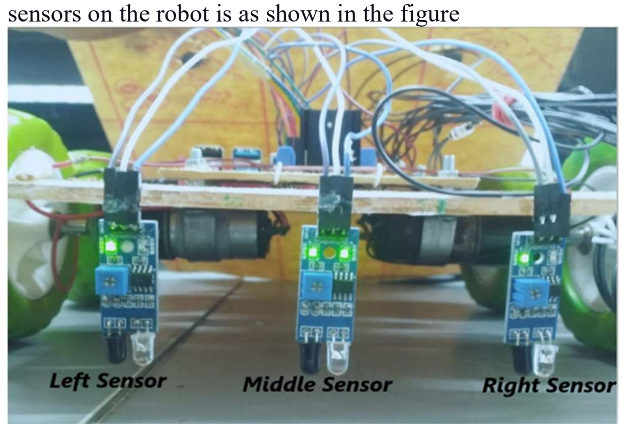

Fig. 15. Sensor Array on robot

Path in the library will look like as follow

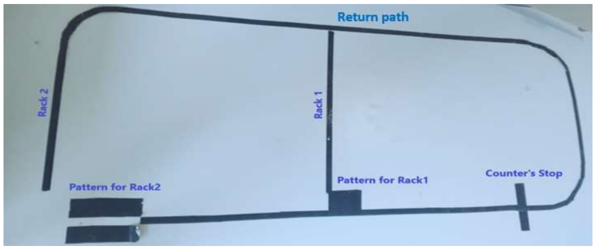

Fig. 16. Path in library

Algorithm is that, if we have three sensors and each can give an output of either 1 or 0, that means we have $2^3 = 8$ combinations of the outputs. Out of these 8 combinations, 4 combinations are responsible for keeping the robot on the black line. The other 4 combinations are for the indication of different racks.

Left sensor Value	Middle Sensor Value	Right Sensor Value	Action
0	0	0	Forward
0	0	1	Turn Left
0	1	0	Move Forward
0	1	1	Rack 1's Path
1	0	0	Turn Right
1	0	1	Rack 2's Path
1	1	0	Rack 3's Path
1	1	1	Stop

1) Obstacle Avoidence

Robot will always follow the black line, but their may be the case if some obstacle comes on the line. Since there would be a designated path in the library whichever obstacle comes in the way is temporary -would leave the path sooner or later.
Robot has Ultrasonic sensors at the front, if they detect any obstacle, it waits until the obstacle is cleared.

C. Working of Arm

Movement of arm is because of stepper motors. Stepper Motor Rotates and lead screw transforms rotatory motion into circular motion and area in the work envelope can be reached. Stepper Motor comes in different sizes, we are using 6 wire Nema23 stepper motor. Basic operation of the motor depends upon the inductive coils which rotates the rotor when energized. Wires coming out of the stepper motor corresponds to one of the windings. Connection of the wires can be identified using, resistance check method. Method is as follows,

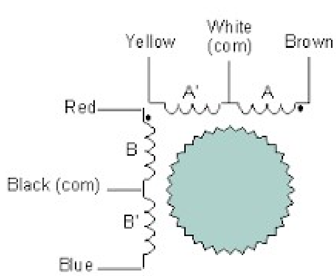

Fig. 17. Boundary detection through Hough Transformation

- Resistance between two wires of different coils is infinite
- In the above figure, Black is the common wire if and only if, its resistance with blue or red is somewhat half of the resistance between blue and red

Nema23 stepper motor rotates to 1.8 degrees per step. So, for a complete rotation it takes 200 steps. Arm is 18 inches, so we can move the arm to the desired location as per the following formula.

$$steps\ needed = \frac{steps\ per\ revolution}{Dimension\ of\ Lead\ Screw}$$

Grippers are controlled using servo motors. Servo motors can rotate with great precisions. We have used the MG996R servo motor for opening and closing the claws of the gripper. Servo motors work on 50Hz PWM signal, where servo's position depends upon the duty cycle of the signal. When the servo motor is commanded to go to certain position, it goes and holds that position. If any external force pushes against the servo, servo will resist. Maximum counter force that a servo can exert is the torque rating of the servo [10]. We have used the servo motor which have 13 kg/cm torque [11]

V. Conclusion

It is evident that *using spine labels instead of RFID tags* to find position of books is a better solution for library robots. It has the following advantages

- No need to place separate identifiable tags (RFID) on each book
- No need to save location of all the books in a database
- No separate hardware setup is required as in RFID based systems
- Multiple books can be processed together, while RFID based system can process only one book at a time

Moreover, *Modified Line Following Algorithm* can easily access many racks by the recognition of unique patterns on the path. This algorithm is highly scalable.

Number of Racks Accessible = $2^{(number\ of\ IR\ sensors)} - 5$

We are using 3 IR sensors hence can access 5 racks. If someone wants to access 123 racks in the library, he will only need 7 IR sensors. So, in this algorithm one can easily modify the number of sensors to access more racks.
We have made a prototype that can place books in the second shelf of the book. In future, this technology and the algorithms can easily be scaled to place books in all of the racks and shelves.

VI. Refrences

[1] M. Axelsson, "The little robot that lived at the library," Medium, 12-Sep-2019. [Online]. Available: https://towardsdatascience.com/the-little-robot-that-lived-at-the-library-90431f34ae2c. [Accessed: 24-Jan-2022].

[2] S. Magazine, "This robot librarian locates haphazardly placed books," Smithsonian.com, 14-Jun-2016. [Online]. Available: https://www.smithsonianmag.com/smart-news/robot-librarian-locates-haphazardly-placed-books-180959381/. [Accessed: 24-Jan-2022].

[3] “Library assistant robot - IJERT.” [Online]. Available: https://www.ijert.org/research/library-assistant-robot-IJERTV8IS060283.pdf. [Accessed: 01-Feb-2022].

[4] “Development of a self-navigating algorithm for Library Book Finder Robot,” IEEE Xplore. [Online]. Available: https://ieeexplore.ieee.org/document/8275140/. [Accessed: 01-Feb-2022].

[5] M. Nevetha and A. Baskar, "Automatic Book Spine Extraction and Recognition for Library Inventory Management", Proceedings of the Third International Symposium on Women in Computing and Informatics - WCI '15, 2015. Available: 10.1145/2791405.2791506 [Accessed 24 May 2022].

[6]"OpenCV: Canny Edge Detection", Docs.opencv.org, 2022. [Online]. Available: https://docs.opencv.org/4.x/da/d22/tutorial_py_canny.html. [Accessed: 24- May- 2022].

[7]"Hough Line Transform — OpenCV-Python Tutorials beta documentation", Opencv24-python-tutorials.readthedocs.io, 2022. [Online]. Available: https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_houghlines/py_houghlines.html. [Accessed: 24- May- 2022].

[8] "Build Optical Character Recognition (OCR) in Python", Medium, 2022. [Online]. Available: https://towardsdatascience.com/build-optical-character-recognition-ocr-in-python-28d1c7b77da3#:~:text=In%20Python%2C%20we%20can%20do,Hewlett%2DPackard%20as%20proprietary%20software. [Accessed: 24- May- 2022].

[9] "Build Optical Character Recognition (OCR) in Python", Medium, 2022. [Online]. Available: https://towardsdatascience.com/build-optical-character-recognition-ocr-in-python-28d1c7b77da3#:~:text=In%20Python%2C%20we%20can%20do,Hewlett%2DPackard%20as%20proprietary%20software. [Accessed: 24- May- 2022].

[10] "Build Optical Character Recognition (OCR) in Python", Medium, 2022. [Online]. Available: https://towardsdatascience.com/build-optical-character-recognition-ocr-in-python-28d1c7b77da3#:~:text=In%20Python%2C%20we%20can%20do,Hewlett%2DPackard%20as%20proprietary%20software. [Accessed: 24- May- 2022].