Drawbot: A Mobile Robot for Image Scanning and Scaled Printing

Jaydeep Saha, Tejas Niphadkar, and Arpan Majumdar
National Institute of Technology, Warangal, India
Email: {jaydeepsaha1995, tejas.niphadkar, arpan.gvdn7}@gmail.com

***Abstract*—This paper proposes mobile robots with scanning and plotting capabilities as a solution towards printing large two dimensional designs. In this respect, maintaining accuracy of the design during reproduction is an important factor. This paper proposes a novel two-step approach to design such robots to serve the purpose. The first step involves an image processing technique which converts an input design/image into an *m x n* matrix of desired size and resolution. The second step involves feeding this information to a micro-controller that controls the movement of the robot on the subjected area of reproduction. We elaborate on how robot's movement and plotting can be simultaneously controlled using our approach. The proposed approach is further verified by building a prototype and subjecting it to various pattern designing tasks. The simplicity of the design and flexibility in plotting can be useful to produce complex figures on a given area.**

***Index Terms*—binary image, gray-scale, image processing, microcontrollers, serial communication**

I. Introduction

A large section of robots nowadays is used to ease the human effort or increase accuracy [1]-[3]. Considering situations where markings in parking areas or fields have to be made, it is difficult to paint either humanly or operate bulky machines. Even printing huge billboards using printers are cumbersome as the printing process has to be fragmented. This necessitates the development of a solution which can simplify these operations on large areas. A robot which can perceive a design shown to it and accurately reproduce it on any given area as directed is a solution. Development of algorithms and robot design that takes into consideration the flexibility of the size of the image to be printed is the primary objective.

The idea behind this paper is to convert the captured image to a binary image of *m x n* pixels matching with the desired resolution of the actual design to be produced on an *m x n* area. The matrix so obtained is fed to the microcontroller which decides the movement of the bot as well as its drawing action.

A simple methodology is developed which takes care of the flexibility of the dimensions of the image as well as the ease of implementation.

Manuscript received July 25, 2015; revised December 15, 2015.

II. Related Work

Since the birth and death of Senster [4], the large scale Robotic creatures developed by Ed Inhatowicz [5] and exhibited in the early 1970s at Phillips' Evoluon, the former science museum in Eindhoven, a significant population of robotic entities have entered the art world.

Patrick Tresset's artist robot Paul performs series of organized scribbles to produce a portrait [6]. This presents efforts in implementing two different versions of visual feedback to permit the robot to iteratively augment and improve a drawing which is initially built from a process of salient lines recovery. The first form of visual feedback involves a purely internal (memory-based) representation of regions to render via shading by the robot. The second version involves the use of camera as an 'eye', taking new snapshots of the artifacts in progress. This is then analyzed to take decisions on where and how to render next shading.

Research Disney's latest creation is the turtle shaped 'BeachBot', an autonomous robot that draws picture on the beach using state-of-the-art technology [7].

But, these do not provide the flexibility of reproducing a design of the desired size. Some restrict itself to the size of a hand drawn portrait while others aim at drawing large pictures e.g.: on sand.

Graph based approach to image processing is intended for use with images obtained from sensors having space variant sampling grids [8]. Using an expanded Connectivity Graph called the Transformation Graph, efficient algorithms can be implemented for matching log map images and solve the template matching problem for space variant images.

Image Processing based robotic systems have been used for applications like "Auto Inspection System Using a Mobile Robot for Detecting Concrete Cracks in a Tunnel" [9] or "Robot Assisted Tiling of Glass Mosaics with Image Processing" [10].

III. Proposed Approach

A. Design

The prototype that was designed was simple consisting of:

- 2 X Arduino Uno boards
- Servo Motors
- 2 X 1000 rpm motors

- Chassis and wheels
- Webcam
- Motor Controller IC
- Markers

To give an appearance of a Humanoid Robot, the webcam was mounted at top. One of the Arduino boards was mounted at the bottom along with the motor controller IC on the chassis above the wheels. It was used to control the movement of the robot as well as communicate with the other Arduino. Another Arduino board was mounted at the back that was meant to control the drawing arm action while in constant communication with the bottom Arduino.

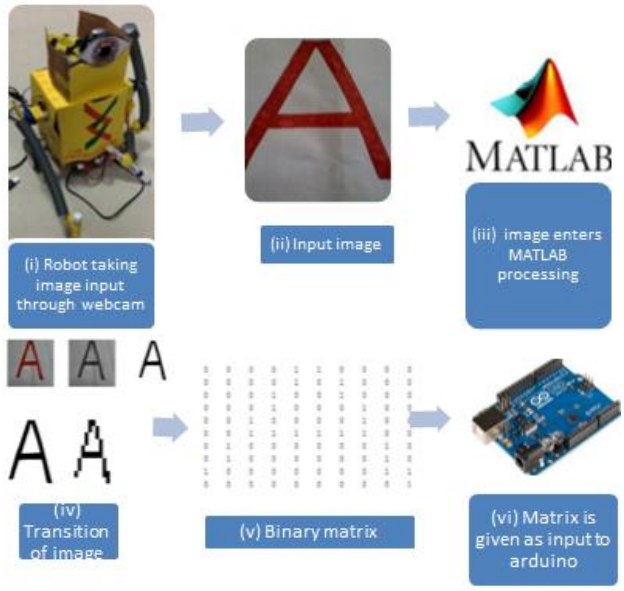

Figure 1. The steps involved in processing the input image before the Arduino starts processing.

B. Methodology

Step 1: Image Processing

The image processing feature of MATLAB provides a comprehensive set of reference that includes standard algorithms, functions and apps for image processing, analysis, visualization and algorithm development [11], [12]. This tool was made use of and a simple algorithm was developed as depicted in Fig. 1.

- Capturing a snapshot
- Converting RGB image to grayscale
- Converting grayscale image to binary image using proper threshold
- Filtering the noise i.e. blobs with size less than 100 pixels are removed
- Crop the image to appropriately fit into its bounding rectangle
- In the "lines" method explained in step 3, the time of plotting can be minimized by choosing the direction of printing that needs to cover lesser number of lines. This can be accomplished by traversing the binary matrix and calculating the connected components, i.e. the groups of connected '1's in each row. An algorithm is then implemented which compares between various orientations and selects the orientation that would consist of lesser number of connected components. This is depicted in Fig. 2. The binary matrix is then transposed accordingly.
- Converting the binary matrix into usable form, i.e. Matrix conversion or flipping of binary rows and writing the binary matrix into a text file.

Figure 2. Minimisation of connected components by transposing binary matrix to optimise printing through lines algorithm.

Step 2: Exporting Binary Matrix to Arduino

The binary matrix obtained after matrix conversion was so arranged that it provided information procedurally to the bottom Arduino during the bot's traversal while plotting. This binary information was read into the Arduino IDE using Processing IDE.

Step 3: Implementing the Algorithms Using Arduino IDE

'**Arduino 1**' is regarded as '**Motion Controller Arduino**' whereas '**Arduino 2**' is regarded as '**Printer Arduino**' in the following algorithms:

- **'Dots'**: The algorithm involving plotting of dots to evolve a design, allows 'Arduino 1' to drive the bot unhindered in a straight path when there are '0's in the binary information. Whenever a '1' is encountered, a digital pin is made high which is connected to 'Arduino 2' which directs it to execute its code, whereas 'Arduino 1' waits for signal from the 'Arduino 2' before executing the rest of its code. 'Arduino 2' which controls hand servos completes its action of plotting and sends a similar signal to 'Arduino 1' as a message of completion of its work. After every '*m*' pixels/positions, the bot shifts itself laterally by 1 pixel to print the next row.
- This process continues until the traversal through *m x n* positions is complete.

Pseudo Code -

1. For Arduino 1:

```
#set counter
c=1; a [m x n]; i=0;
rs(receiving); ts(transmitting);
While 1 do
For  i<m x n then
  i++

  # for odd lines
    If c%2!=0 then
      If a[i]==0 then
```

```
        drive the wheel motor
      End If
      If a[i]==1 then
       stop the bot
       write 'ts' pin HIGH
        While  !read 'rs' pin == HIGH  do
          keep the motor input LOW
        End While
        drive the wheel motor
      End If
      If  (i+1)%m==0  then
       laterally shift robot 1 pixel
       c++
      End If

  #for even lines
   Else
      If a[i]==0 then
       drive the wheel motor
      End If
      If a[i]==1 then
       stop the bot
       write 'ts' pin HIGH
        While  !read 'rs' pin == HIGH  then
          keep the motor input LOW
        End While
        drive the wheel motor
      End If
      If  (i+1)%m==0  then
        laterally shift robot 1 pixel
        c++
      End If
    End If
  End For

  End While
```

2. For Arduino 2:

```
rs(receiving); ts(transmitting);
While 1 do
  set hand servos to default angle
  write 'ts' pin LOW
    If read 'rs' pin==HIGH  then
      rotate hand servos to mark a dot
    End If
 write 'ts' pin HIGH
 provide delay
 write 'ts' pin LOW
End While
```

- **'Lines'**: The other algorithm that was used is almost similar to the previous one. Whenever, there are appearance of '1's, the drawing action continues, that is 'Arduino 1' keeps the marker lowered. However, when a '0' is encountered the 'Arduino 1' sends a signal to 'Arduino 2' which directs it to lift the marker up or suspend the plotting action. The bot shifts similarly by 1 pixel after every 'm' pixels/positions.

Pseudo Code-

1. For Arduino 1:

```
#set counter
c=1;  a [ mxn ]; i=0;
rs (receiving); ts(transmitting);
While 1 do
For  i< mxn  then
  i++

 #for odd lines
   If  c%2!=0  then
    If  a[i]==0  then
     write 'ts' pin HIGH
      While  !read 'rs' pin==HIGH  do
        stop wheel motor
      End While
    drive the wheel motor
    End If
    If a[i]==1  then
     write 'ts' pin LOW
      While(!read 'ts' pin==HIGH)
       stop wheel motor
      End While
    drive the wheel motor
    End If
    If  (i+1)%m==0  then
      write 'rs' pin HIGH
      shift the robot by 1 pixel laterally
      c++
    End If

 #for even lines
   Else
     If  a[i]==0  then
      write 'ts' pin HIGH
       While  !read 'rs' pin==HIGH  do
         stop wheel motor
       End While
     drive the wheel motor
     End If
     If a[i]==1  then
      write 'ts' pin LOW
       While(!read 'ts' pin==HIGH)
        stop wheel motor
       End While
      drive the wheel motor
     End If
     If  (i+1)%m==0  then
```

write ‘rs’ pin HIGH
shift the robot by 1 pixel laterally
c++
End If
End If
End While

2. For Arduino 2:

rs(receiving); ts(transmitting);
While 1 **do**
set hand servos to default angle
write ‘ts’ pin LOW
If read ‘rs’ pin==LOW
rotate the hand servos to plot
write ‘ts’ pin HIGH
provide delay
write ‘ts’ pin LOW
Else
keep the hand lifted
write ‘ts’ pin HIGH
provide delay
write ‘ts’ pin HIGH
End If
End While

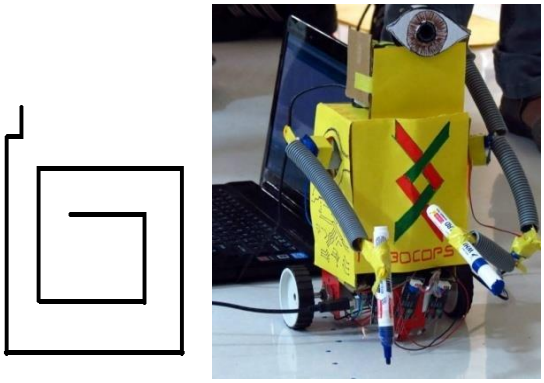

Figure 3. The Robot-prototype plotting the design shown during final test

TABLE I. TIME TAKEN BY PROTOTYPE TO PLOT USING THE TWO ALGORITHMS

Algorithm	**Experiment**		
	Design	***Drawing Area***	***Time (approx)***
(i) dots	Spiral design	1m x 1.5m	2.5 min
(ii) line	Solid letter ‘A’ (5cm thick)	1m x 1.5m	2 min

IV. CASE STUDY

The prototype developed is tested for its performance while it implements various designs ranging from simple parallelograms and solid alphabets to layout of houses. Fig. 3 shows the bot while plotting a spiral design using the ‘dots’ algorithm. The time duration of plotting using different algorithms are also observed. An idea of the time taken by the prototype during designing can be obtained by observing the values tabulated in Table I.

V. RESULTS AND ANALYSIS

The case study shows that the time taken while plotting using ‘dots’ method consumes slightly more time compared to the ‘lines’ method. However, the ‘dots’ method is seen to provide higher accuracy.

Time taken by the robot as well as energy consumption is directly proportional to the number of dots in the ‘dots’ method whereas number of lines in the ‘lines’ method. Number of pixels in a figure cannot be altered, however, in ‘lines’ method the traversal of the bot can be minimized by calculating the connected components or groups of ‘1’s in each row of the final binary image and then deciding on the orientation of printing as shown in Fig. 2. This optimization when implemented is observed to reduce the plotting time (reduce drastically in few cases).

Therefore, it is suitable to apply ‘dots’ method for designs that need higher amount of accuracy and are smaller in size. Whereas, the ‘lines’ method may be used in applications that involve large design and accuracy is not of utmost importance.

However, the time required can be reduced further by using better mobility of the finally manufactured bot.

VI. CONCLUSION

The proposed solution for reduction in human toil for producing considerably large two dimensional designs uses the image processing power of MATLAB and processing technology of Arduino to build a robot that upon perceiving a relatively smaller design on paper can reproduce it on a larger area of any size. The tradeoff is the time and the amount of paint used. The designer is given the freedom to use his wisdom to decide upon the size of the matrix that he will be using for plotting the design on a given area taking care of the clarity of the design as well as the cost and the plotting time considerations. The flexibility provides the possibility to diversify the usage of the robot.

This Robot is estimated to perform very effectively in applications like sports field markings, markings in parking areas, etc. The proposition is economic considering the fairly inexpensive components used compared to the designs that it is able to perform. The energy requirement of the robot is considerably low as the whole prototype was operated using three 9V batteries.

There is scope of development of the algorithms used in the prototype and they can be implemented according to the design to be made. Even intelligence can be developed which allows the robot to choose the optimized plotting algorithm.

ACKNOWLEDGMENT

The major part of the prototype was developed at Hackathon 3.0 organized by the Lakshya Foundation.

The author sincerely wishes to thank his Team members Tejas Niphadkar, Arpan Majumdar, Aniruddha

Bala, Shubham Dhanuka and Jyotik Parikshya for their sincere efforts in nurturing the ideas and subsequent development of the prototype. He also extends deep gratitude towards Surya Penmetsa for his expert mentorship.

REFERENCES

[1] A. R. Beasley and D. R. Howe, "Increasing accuracy in Image guided robotic surgery through tip tracking and model based flexion correction," *IEEE Transactions on Robotics*, pp. 292-302, February 2009.

[2] L. Kevin Conrad, S. P. Shiakolas, and T. C. Yih, "Robotic calibration issues: Accuracy, repeatability and calibration," in *Proc. 8th Mediterranean Conference on Control & Automation*, Rio, Patras, Greece, July17-19, 2000.

[3] K. Fischer, C. L. Jensen, and F. Kirstein, Designing for Ease of Collaborative Effort in Human-Robot Joint Action.

[4] A. Zivanovic, "The development of a cybernetic sculptor: Edward Inhatowicz and the senster," in *Proc. Fifth Conference on Creativity and Cognition*, New York, 2005, pp. 102-109.

[5] F. Ghedini and M. Bergamasco, "Robotic creatures: Anthropomorphism and interaction in contemporary art," in *Proc. IEEE Ro-Man International Symposium on Robot and Human Interactive Communication*, 2010, pp. 731-736.

[6] P. Tresset and F. F. Leymarie, "Portrait drawing by paul the robot," *Computers and Graphics*, February 2013.

[7] Disney Robot Draws Giant Sketches on the Beach. (January 8, 2015). [Online]. Available: http://spectrum.ieee.org/automaton/robotics/robotics-hardware/disney-research-beachbot

[8] S. R. Wallace, P. W. Ong, B. B. Bederson, and E. L. Schwartz, "Space variant image processing," *International Journal of Computer Vision*, vol. 13, no. 1, pp. 71-90, September 1994.

[9] S. N. Yu, J. H. Jung, and C. S. Hang, "Auto inspection system using mobile robot for detecting concrete cracks in a tunnel," *Automation in Construction*, vol. 13, no. 3, pp. 255-261, May 2007.

[10] B. Kaya, A. Berkay, and F. Erzincanli, "Robot assisted tiling of glass mosaics with image processing," *Industrial Robot: An International Journal*, vol. 32, no. 5, pp. 388–392, 2005.

[11] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley, 1992.

[12] Image Processing Toolbox. [Online]. Available: http://in.mathworks.com/products/image/

Jaydeep Saha is a student pursuing pre-final year of Bachelor degree (B.Tech) in Electrical and Electronics Engineering at National Institute of Technology, Warangal, India. He is the Technical Secretary of IEEE Student Body, NIT Warangal.
His Interest areas are Embedded System and Power Electronics.

Arpan Majumdar is currently pursuing pre-final year of B.Tech. in Electrical and Electronics Engineering at National Institute of Technology, Warangal, India. His areas of interest include image processing, modeling and simulation of electrical machines and automation of electrical systems.

Tejas S. Niphadkar is currently pursuing pre-final year of B.Tech. in Electrical and Electronics Engineering at National Institute of Technology, Warangal, India. His areas of interest include embedded system, algorithmic coding and web development.