Designing a Tactograph

S R V Vishwanath, S Veer, R Mittapally, N Chandrachoodan, A Prabhakar, S Srinivasan

***Abstract*—We have developed a low cost assistive device for tactiling simple images in pre-literacy and early literacy books for visually impaired children. A parallel five bar manipulator was chosen after comparing parameters like: cost, manufacturability, repeatability etc. The correction pen was found appropriate for fluid dissemination. A parallelogram linkage was designed for the actuation of the correction pen. Link lengths of the parallel five bar manipulator were chosen based on the workspace area and resolution using Matlab. Adams simulation was used to determine the torque requirement and Abaqus was used to analyze the link deflections. A PIC18F4550 microcontroller with USB support was used for controlling the motors. The host side software was completely written in python. Linear interpolation algorithm was implemented for end effector control. A GUI was made to enable ease of use for end user.**

I. Introduction

Reading is an essential and fundamental need for survival in an interconnected world. However, in India, there is a dearth of affordable tactile books for children who are in the early stages of developing literacy. Consequently, children who join primary school have poor reading skills and are at a disadvantage compared to their peers. Teaching Braille to young visually impaired children is a challenging task. Few children enjoy reading text which is the reason why children's books are adorned with images to instill an interest towards them. Books with tactile images bring value to both the sighted and the visually impaired child, thus reinforcing the concept of inclusive product design. Exploring such books can be beneficial for these children in multiple ways [1].

- Develop imagination and creativity
- Discover the objects in different images
- Develop the proper representation through the relation among object-image-word
- Develop the exploration of images
- Develop the visual and the tactile sensitivity as a primordial step for writing and learning

Haptic perception of images can be achieved by tactiling or embossing. Embossing being a costly process is normally not preferred by book publishers. Tactiling using fabric paint on already printed material was demonstrated [2] and has been adopted by Sarva Siksha Abhiyan (SSA) in Tamil Nadu. However, the process is manual and time consuming. We have now prototyped a Tactograph, which will empower a small or medium enterprise (SME) to undertake the tactiling of already printed material available in bookstores. Note however that the Tactograph is not suitable for all graphic images. The primary aim is to make pre-literacy and early-literacy books available to the visually impaired. As such, the choice of the book and its images plays an important role in making reading materials accessible, e.g. images with too much detail or non-descriptive text accompanying the image are poor choices. Simple lines and curves are conducive to easy tactiling, and any minimal, but descriptive text, can be added on in Braille while still retaining the original image and text.

II. Methodology

The primary objective of our prototype is to demonstrate the tracing of a predetermined set of lines with a quick drying fluid, resulting in a tactile outline of the image. A sample of a tactile image is shown in Figure 1. The lines to highlight and the colours to use are chosen by an artist. Thereafter, the image is digitized and broken down to a set of curves that the tactograph must follow. This simple approach helps minimize the cost of the tactograph, while the burden of faithful reproduction on multiple copies of the book is taken on by a computer. To achieve these objectives, we had to develop an automated way of correcting for any shifts (translation and rotation) when a fresh image is placed on the tactograph, and then have a mechanical fixture capable of tracing the digitized outline.

SRVV, NC and AP are with the Dept. of Electrical Engineering, and SV, RM and SS are with the Dept. of Mechanical Engineering, Indian Institute of Technology Madras, India, e-mail: vishwa.hyd@gmail.com

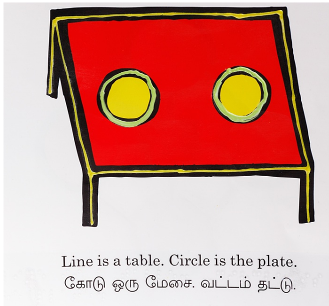

Figure 1. An image tactiled using fabric paint in the book Line and Circle, by Tulika Publishers. The simplicity of the outlines and the text make the book accessible to a child with visual impairment.

A. *Mechanical Design*

There are many ways of designing a plotter. We evaluated three different designs, keeping various technical and economic parameters in mind. These are summarized in Table I, and we conclude that a five bar pantograph is the preferred option.

Table I
Comparative evaluation of different plotter mechanisms.

Parameters	**X-Y Table**	**Two Link Robotic Arm**	**Five Bar Pantograph**
Cost	-	-	+
Maintainance	-	+	+
Simplicity	-	+	+
Manufacturing time	-	+	+
Weight	-	+	+
Integration	-	+	+
Repeatability	+	+	+
Static actuators	-	-	+
Uniform $\Delta x, \Delta y$	+	-	-
Score	**-5**	**+6**	**+8**

A question that arises is, "why can a printer not be re-engineered and used?" A printer plots by scanning the x axis, keeping y constant and this is done for all y. However, the images to be tactiled demand that there be continuity in the plotted points, so that the fluid forms a neat curve. Hence the plotter cannot be made to scan the whole page. Besides, most printers use a drum roller feed mechanism to feed the paper into the printer, which will smudge the white fluid on the paper. We compared two possible mechanisms to control the fluid flow, which we summarize in Table II. The table clearly displays the advantages of using a fluid correction pen for tactiling over a miniature fluid pump with a nozzle. In a similar exercise, we also compared different plotter head mechanisms in Table III, and finally chose to use a programmable five bar pantograph linkage with a whitener pen, for the current prototype. This whitener is mobilized in its up-down motion using a parallelogram linkage. Figure 2 shows the side and top view of the present design, along with a closer look at the plotter mechanism that holds the fluid correction pen.

Table II
TACTILERS COMPARISON

Parameter	Fluid pump	Fluid correction pen
Low cost	-	+
Low maintainance	-	+
Simple manufacturing	-	+
Low weight	-	+
Ease of integration	+	-
Good availability in market	-	+
Score	**-5**	**+5**

Table III
COMPARISON OF PLOTTER HEAD MECHANISMS.

Parameters	Slider crank mechanism	Rack and pinion mechanism	Parallelogram linkage
Low cost	-	-	+
Low Maintainance	-	-	+
Manufacturing simplicity	-	-	+
Manufacturing time	-	-	+
Low weight	-	-	+
Ease of assembly on plotter	-	-	+ Low weight
Score	**-6**	-6	**+6**

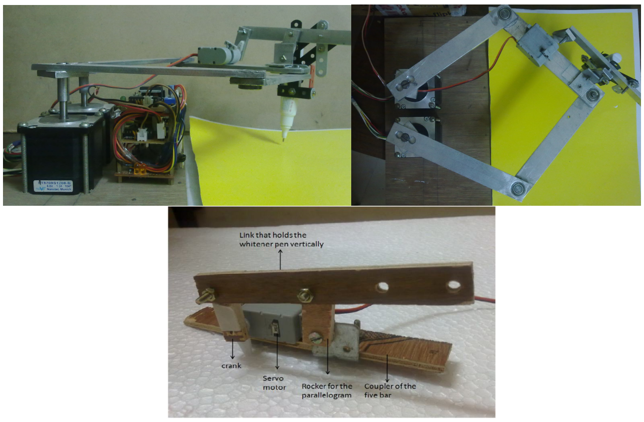

Figure 2. Side and top view of the prototype (first version) that has been designed, with a closer view of the plotter mechanism.

1) Optimization: The optimization of the five bar pantograph requires that we establish appropriate trade-offs between link lengths, distance between the centres of the motor shaft and the distance of an edge of the A4 sheet from the line joining the centre of the motors, as shown in Figure 3. The objective functions of interest are the maximum area (or workspace) on the A4 sheet, and the resolution between points on the workspace. By varying the control variables and following the constraints, several workspace plots were obtained and the best one of them was used to determine the link lengths, ground length and the distance of the A4 sheet from the motors. The workspace plots were generated by writing a Matlab code, with a typical result shown in Figure 4. One of the limitations of the five bar pantograph is that the points in the workspace are not equally spaced. However, the resolution we can achieve is sufficient for our present requirements.

To ensure that the workspace generating code is giving the correct points, a kinematic simulation of the pantograph was also undertaken. This was accomplished, with reference to the diagram in Figure 5, by holding crank2 static and running the rest of the linkage, i.e. crank1, coupler1, coupler 2 and the imaginary ground like a four bar mechanism [3]. After every complete run of the fourbar, the crank2 angle was changed and the fourbar with a new imaginary ground was run again. The singularities in a parallel five bar manipulator have to be avoided while operation since it leads to high torque. Our algorithm will avoid singularities caused by the toggle of the two couplers and one crank attached to them, and the toggle of crank1 r2 and coupler1 r3, i.e., when they align in a

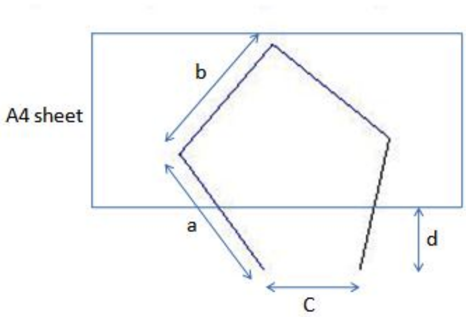

Figure 3. Schematic of the five bar pantograph showing the variables used during constraint optimization.

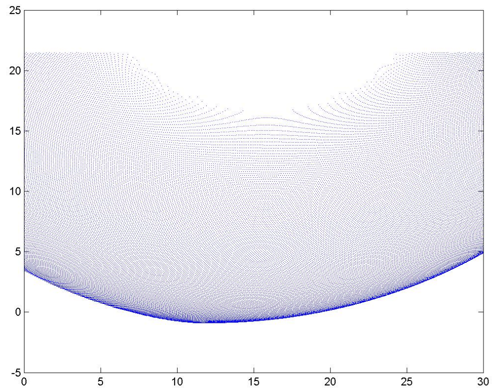

Figure 4. Plot of reachable points on a A4 sheet of paper.

straight line. Similarly, the other toggles considered were the crank2 r5 and coupler2 r4, and the coupler-coupler toggle.

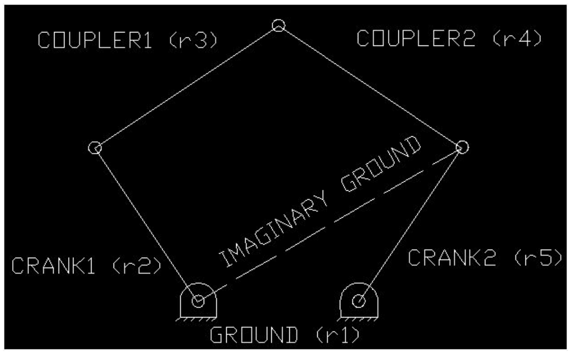

Figure 5. Simulation diagram used to identify and eliminate singularities.

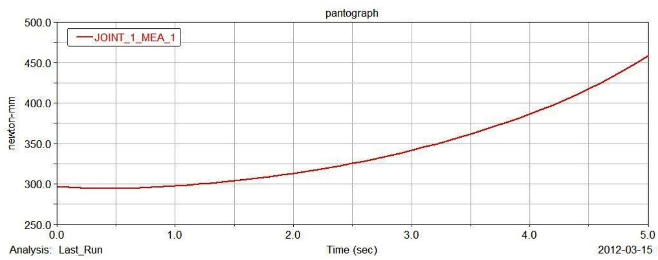

Figure 6. Torque vs time plot generated to estimate motor torque requirements using Adams.

2) Mechanical Loads: The uniformity of our plotter is affected by the load of the plotting mechanism. We used the Adams Multibody Dynamics Simulator to determine that the maximum torque at the joint connected to the motor is around 460 N-mm as observed from the plot in Figure 6. Hence stepper motors with a maximum torque rating of 1.1 N-mm were chosen, giving a safety factor of around 2.4. Similarly, the load on the links of the pantograph were

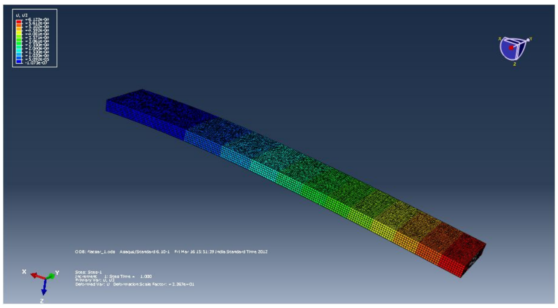

Figure 7. Cantilever beam FEM model.

expected to deform due to the weight of the plotting mechanism at the end of the alumnium flat bars. The links were modeled as cantilever beams in the FEM model using Abaqus to find the maximum deviation under loading. The estimated tip loading on the link considering weight of the succeeding link is around 205 gf, with a moment of 0.26 Nm in a static structural analysis. The end deflection of the beam was estimated to be 0.211 mm, which was verified using a FEM model in Abaqus, shown in Figure 7.

B. Electronics

The electronic design was divided into two major groups, communications over USB and power supply. The USB module acts as an interface between the computer and the power board, while the power module drives the stepper motors and the servo motor.

1) USB Module: There is no dearth of choice when it comes to microcontrollers. As with the mechanical design, we did a comparative evaluation between ATmega16 and PIC18F4550. The comparison in Table IV helps understand why the USB module is built around the microcontroller PIC18F4550 by Microchip. The PIC microcontroller is USB 2.0 compliant, can operate in both low speed (1.5Mbps) and high speed (12Mbps) with 1 Kb Dual Access RAM, has 32 Kb flash memory and needs a 20 MHz external oscillator such that the PIC can be clocked at 48 MHz, using an internal phase locked loop (PLL) division.

Table IV
MICROCONTROLLER ANALYSIS

Processor	Atmega16	PIC18F4550
Number of pins	40	40
Maximum operating frequence	16MHz	48MHz
Maximum I/O pins	32	30
Maximum Program Memory	16KB	32KB
USB support	Through USBtiny	Native
libUSB support	yes	yes
Free IDE/compiler	yes/yes	yes/no
USB Bootloader	-NA-	yes

The number of I/O pins is not a matter of concern, since we use only 6 output pins. The native USB support for PIC18F4550 along with standard library available for USB framework makes it ideal for the present project. However, the compiler for PIC microcontroller is not free but student evaluation version. It was be enough for the project, since the code is not very big. A possible improvement would be to change to PIC18F2550 to save space. An external 20MHz crystal oscillator is used as clock for the device. The device is Bus powered, meaning, it wouldnt require any external power to control the microcontroller. Two status LEDs have been used to represent USB power and proper functioning of the device.

a) Construction of USB device: The device has been constructed based on PIC to PC communication [4]. The schematic shown in Figure 8will help to understand the circuit better. Only the PDx LED has been retained from the circuit given to keep the device minimal. Care has to be taken to ensure that the capacitor at Vusb is not excluded, lest it will give 'Device not found' error. The pins PA0, PA1 are the clock and the direction bit for the first motor respectively. Similarly, PA2 and PA3 are the clock and the direction bits for the second motor respectively. PB3 acts as the position control bit for the servo motor, controlling the Fluid pen. However, since the least acheivable frequency for PWM is 1 kHz for the microcontroller and the servo requires 50Hz, the servo is controlled by ATtiny85 micro controller. The PB3 output of PIC microcontroller is given as input to the PB3 pin of the ATtiny85 microcontroller. The whole circuit is soldered on a custom PCB made for DIP ICs. SMD construction has been avoided to reduce cost.

The device connects to the computer through a standard USB B type female connector. Also, the microcontroller needs to be programmed only once through ICSP programmer. After that, USB bootloader helps to program it through the USB port itself.

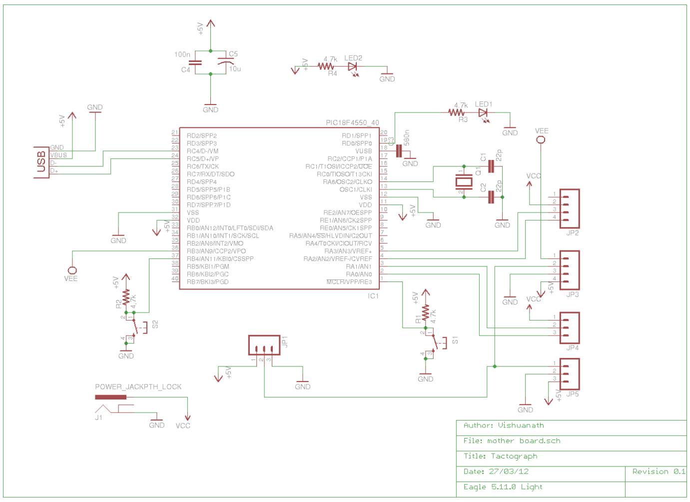

Figure 8. Circuit diagram of USB module.

b) Firmware: The device firmware was written in C, in MPLab IDE and MPLab C18 compiler. The code was written on top of the LibUSB firmware example provided by microchip. The device was configured as low speed device with end point 1 used as the input and output end point. In the basic example provided by microchip, the following changes were made

1) The processIO function was changed to accomodate the interrrupts recieved by the microcontroller
2) The following were the letters used as the interrupts:
   a) $'f'$ - rotate the first motor clockwise
   b) $'r'$ - rotate the first motor anticlockwise
   c) $'c'$ - rotate the second motor clockwise
   d) $'a'$ - rotate the second motor anticlockwise
   e) $'u'$ - Move the fluid pen up
   f) $'d'$ - Move the fluid pen down
3) The device was not configured to return anything to the computer, but it can be reconfigured in case a bulk transfer of the sequence of the rotations is to be sent.
4) The code has been cleaned of all the unneccessary code, for other devices. In case the original code is required, Microchip application libraries can be utilised [5].

2) Power Driver Modules: There are two power driver boards. They are same in all respects except that, one board has servo motor control microcontroller extra. As part of our optimization of costs, we design the circuit to run off a commercially available adpator from Dlink.

a) Servo Motor control: The servo motor requires a 50Hz PWM output for position control. Hence we need a pulse of 20 ms duration to control the motor. Sending a pulse of 1 ms duration keeps the servo at 0^o positon and sending a 2 ms delay keeps the servo at 180^o. Please note that this pulse duration is not very precise and may require some trial and error to find the exact duration of the pulse for the corresponding angle

We used the ATtiny85 for servo control. It is a compact microcontroller with 8 pins, 2 KB of flash memory and an 8 bit timer.

The reason for choosing this microcontroller is for compactness of the circuit.

The pin PB3 from the PIC microcontroller is connected to PB3 pin of the ATtiny85 microcontroller. When PB3 is logic 1, the servo is set at 180° and when PB3 is logic 0, the servo is set at 90°. The circuitry for the microcontroller is shown in Figure 9. The code for this microcontroller is written in **AVRstudio**, with **WINAVR.** The program is burnt with **USBtinyISP**. The improvement for the servo control would be to include the PWM generator in the PIC microcontroller itself, which would save a programming step.

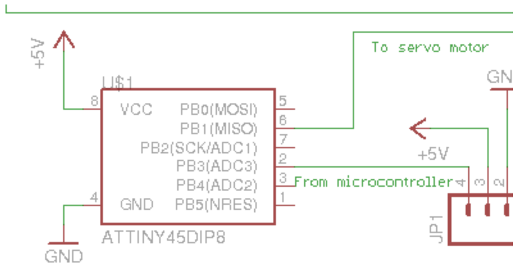

Figure 9. Circuit diagram for servo motor control.

b) Power Board: The stepper motors are rated for 6 V at 1 A current. However, since the torque requirements are low, the motor is run at $5V$. This also ensures that the current is below 1 A and hence the total current requirement stays below 2.5 A.

The interface to stepper motors is a $L297-L298$ pair. The $L297$ is connected to the microcontroller, which gives the clock and the direction. A negative edge causes the $L297$ to change the state. The option for half step or full step is hardware selectable. In future, the step selection will be made software selectable.

The $L297$ also has an inbuilt chopper for current control. A trimport connected to the V_{ref} pin helps to control the current throught the stepper motor windings. For this purpose, an RC network needs to be connected to the OSC pin of the driver.

The four pins of $L297$ output are connected to inputs of $L298$. The $L298$ power driver is a full H-Bridge power driver with maximum current of 2 A per channel. Two 1Ω resistors connected to current sensing pins act as current sense inputs to $L297$. The 1Ω resistor should be of atleast 1 Watt type to avoid burning. However, it is suggested to go for 2-5 Watts. The circuit for one stepper motor driver is shown in Figure 10.

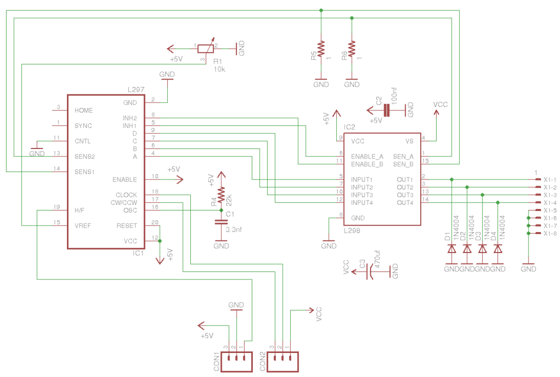

Figure 10. Circuit diagram of the power module.

Table V
ANALYSIS OF CURRENT DRAWN UNDER DIFFERENT OPERATING CONDITIONS.

Motor1	Motor2	Servo motor	Current drawn (A)
2 coils On	2 coils On	Up	2.43
2 coils On	2 coils On	Down	2.16
2 coils On	1 coil On	Up	1.76
2 coils On	1 coil On	Down	1.42
1 coil On	1 coil On	Up	1.20
1 coil On	1 coil On	Down	1.12

A 5 V, 2.5 A power adapter by Dlink is used for power input. The $L298$ chip may get hot during operation but addition of a heat sink is not necessary, since the heat is within threshold. However, if it gets too hot, check all the connections to ensure that the power drawn is not exceeding the chip's power ratings. The power adapter also gets hot during operations, which is normal. However, if it gets too hot, the fuse may blow inside, rendering the adapter useless. Table Vgives a rough estimate of the current drawn by the electronics under various operating conditions.

C. Software Implementation

The device connects to the computer through USB protocol. The host computer communicates through the software. For this purpose, the control program is written completely in python with the help of libusb. Python code is elegant and easy to understand, and comes with the following advantages:

1) Fast prototyping: Since the language is easy to write and implement, free of memory leaks and segmentation faults, it allows us to concentrate more on the logic than handling memory.
2) Large libraries support: For every purpose there is a python modules, ensuring that we need not reinvent the wheel. External libraries that are imported are Scipy (and Numpy), pyusb, python image library, VideoCapture for windows and the v412 library for Linux, and wxpython.
3) Cross platform: The code works both on both Linux and Windows operating systems.

Finally, an executable will be created so that the user installation is easy. To maintain readability and distinction between various jobs, the whole software is split into 4 modules.

1) Data module: As the name suggests, the data module does all the data handling part. The main processes executed by the module include extracting data from the data files, writing them to global data arrays, retrieving coordinates for a given angle pair and vice versa, and generating the sequence of steps to be followed by the motors.

a) Data file format: The data is split into two parts, the coordinates file and the angles file, each consisting of two rows, corresponding to the two motors. In the angles file, the first column corresponds to the right motor, assuming the machine is viewed from the top, the links facing away from the user.

The data, can be provided in an excel form, in which case, the data must be extracted to make the two data files.

2) Motor module: The motor module acts as interface between the host and the device. The module depends on pyusb module. The pyusb module further depends on libusb which creates an access layer for the usb device. The module has functions to send commands to the USB device to move the motors. This includes the servo position toggle functions, and the draw line function. The drawing function uses linear interpolation to go from one point to another. The linear interpolation algorithm is

1) Assuming a distance between points, create N points between the two given points.
2) Find the nearest point to each point on the grid
3) Remove all duplicate points

Also, there are few test functions in the module, which help check the device. Note that in linux, administrative previlages are required, since libusb needs permission to communicate.

3) Image library module (imlib): The imlib module processes input image for tracing on the paper. The module was initially written using python image library but was later changed completely to scipy to improve speed. The module uses the ndimage package from scipy for image processing routines. The main functionalities that have been implemented allow us to digitize a previously tactiled image e.g. that shown in Figure 1.

1) Register class finds the r, θ shift between the initially taken image and the subsequent images. This class has been implemented using FFT based image registration[5] [6]

2) Webcam class takes input from the webcam and processes it to give the edges and the guide image for manual tracing. This class depends on videocapture in windows and v4l2 module in linux
3) Link class links the broken segments in the edge map. The output from canny edge detection gives edges which may be broken at some points. This class links possible broken edges and hence reduces the number of edge segments
4) Canny class does canny edge detection [8]. It is a multi stage, very precise but slow edge detection. The parameters for the class are the sigma for the Gaussian blur, the window size for the Gaussian kernel, the low threshold and high threshold.

The Canny and Link classes will yield the x, y coordinates that make up the lines and curves that we need to trace. When the user places a new sheet with an image that needs to be tactiled, we first correct the x, y coordinates for any translations or rotations using the Register class. Thereafter it becomes a simple task of controlling the motors and the plotter head mechanism.

4) User Interface (pytacto): This module integrates all the above modules and also provides a graphical user interface (GUI) for the software. The GUI helps to track the points the pen is expected to go when using manual plot. The steps followed initially are:

1) Start the device. If not connected, exit.
2) Start the splash screen to give the user feedback while loading files.
3) Load the necessary data files.
4) Switch off the splash screen, start the GUI.

Once the gui starts, the program shows the image taken by the webcam, which will aide in tracing the required curve. The commands for creating segments are:

1) **ctrl+left click** creates a segment from the last created point
2) **shift+left click** creates a new point, disjoint from the last segment
3) **left click** on a point moves the point.

Once the points are clicked, pressing the plot button will start the machine and once the plot is done, the GUI gets back to action again.

III. Cost Analysis

An important aspect of this project is the cost. As has been mentioned earlier, embossing is an expensive proposition and publishers and bookstores are reluctant to produce and stock too many titles with embossed images. A low cost device such as the Tactograph is meant to encourage SMEs or even schools to undertake the tactiling of books based on the demand, of previously published books, thus making low cost tactile books easily available. Tables VI and VII give a summary of the costs involved in making a single unit. We have incurred a total prototyping cost of **Rs 13,040/-** for a single unit. By removing expensive mechanical parts like leadscrew, recirculating ball screw, and timing belts, we have managed a bill of materials (BOM) considerably lower than commercially available x-y plotters. However, it is likely that the final cost price of the Tactograph would increase once we include costs of sales, marketing and after sales support. However, a further reduction in BOM is likely once the production of the Tactograph is transferred to a SME.

Table VI
COST OF MECHANICAL COMPONENTS

S.No	Name	Quantity	Rate	Cost
1	Stepper motors	2	3300/-	6600/-
2	Machining	-NA-	-NA-	3000/-
3	Aluminium	-NA-	-NA-	300/-
4	Bearings	6	30/-	180/-
5	Servo motors	1	600/-	600/-
6	Miscellany	-NA-	-NA-	200/-
Total				**10,800/-**

Table VII
COST OF ELECTRONIC COMPONENTS

S.No	Name	Quantity	Rate	Cost
1	L297	2	150/-	300/-
2	L298	2	430/-	860/-
3	PIC18F4550	1	350/-	350/-
4	ATtiny85	1	100/-	100/-
5	Connectors	-NA-	-NA-	100/-
6	Analog circuitry	-NA-	-NA-	100/-
7	Power adapter	1	250/-	250/-
8	PCB and boxing	-NA-	-NA-	100/-
Total				**2160/-**

IV. Results and Summary

The efforts spent on mechanical optimization are well rewarded. This is evident from the images in Figure 11 where we can clearly see a faithful reproduction of the original using a pen. However, further optimization is needed to achieve the clarity required with the correction fluid. The last (right) image shows how we are able to create simple tactile objects, but the locations where we start and end the curves shows a smear of fluid. We are currently working on a new prototype that aims to fix this problem.

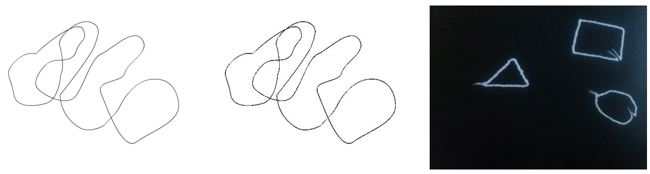

Figure 11. Original (left) and Tactograph reproduction (middle) of a pen sketch. Simple shapes with the correction fluid (right).

References

[1] R. Cziker, “The exploration of tactile images”, ICEVI World Conference, (p. 8). Kuala Lumpur, 2006.
[2] Word Wise, Retail Plus, The Hindu, 01 Feb 2009.
[3] R.L. Norton, Kinematics and Dynamics of Machinery, New Delhi:Tata McGraw Hill, 2009.
[4] EE 494-Engineering Design II, Implementation of a USB PIC-to-PC communication, METU-EEE.
[5] Microchip Application libraries. www.microchip.com/MAL
[6] Christoph Gohlke. “FFT based image registration using Python”. Internet:http://www.lfd.uci.edu/∼gohlke/code/imreg.py.html, 2011.
[7] B. Srinivasa Reddy and B. N. Chatterji, “An FFT-Based Technique for Translation, Rotation, and Scale-Invariant Image Registration”, IEEE Trans. Image Processing, **5**, 1996.
[8] Sohaib A. Khan. “Canny Edge Detector: Implementation”. http://suraj.lums.edu.pk/∼cs436a02/CannyImplementation.htm, 2002.