Building a Laser Projector From Scratch

Alex Hornstein

Our lives are surrounded with data. Emails, phone calls, credit card transactions– our actions generate countless data, all of which are stored in computers, smart cards, CDs, flash drives...the list goes on. But these data are no good to us if we cannot see and interact with it. And while the cost of storage devices and computers has decreased drastically as their portability and availability have increased, the devices which display the data have become the bottleneck.

Display devices are bulky, expensive, and costly. CRTs, the most common display device, can weigh thirty or forty pounds, use hundreds of watts, and have intricate control electronics. LCDs, while lighter, are still delicate and full of intricate driving electronics. LCD projectors consume hundreds of watts, and are very delicate, as well as having very involved driving electronics.

While portable computers, such as PDAs, use LCD screens, the screens use a large portion of the devices' power, and are the limiting factor in reducing the devices' size. Reducing the screen size may increase a device's portability and power consumption, but it is a trade-off with ease of use, since the screen can become too small to use easily.

This leaves a hole that can be filled with a projector. If a projector can be built small enough, then the size of computing devices can be reduced drastically, essentially to the size of the projector, and the display can be on any nearby surface. Current LCD technology does not have high enough resolution in an LCD screen to make a small-scale projector, and LCD projectors also require several focusing mirrors as well as an intense light source. In addition, the light source's intensity decreases according to the inverse-squared law, meaning that as the surface being projected on moves farther from the projector, it gets quadratically dimmer.

There is another type of projector that has not been used as much as LCD projectors. This is a laser projector. It projects by taking a single laser beam, and using mirrors, sweeping it across a display plane. This projector can be made as small as the mirrors and the laser. This project is a demonstration that such a projector can be designed and built robustly, cheaply, and effectively.

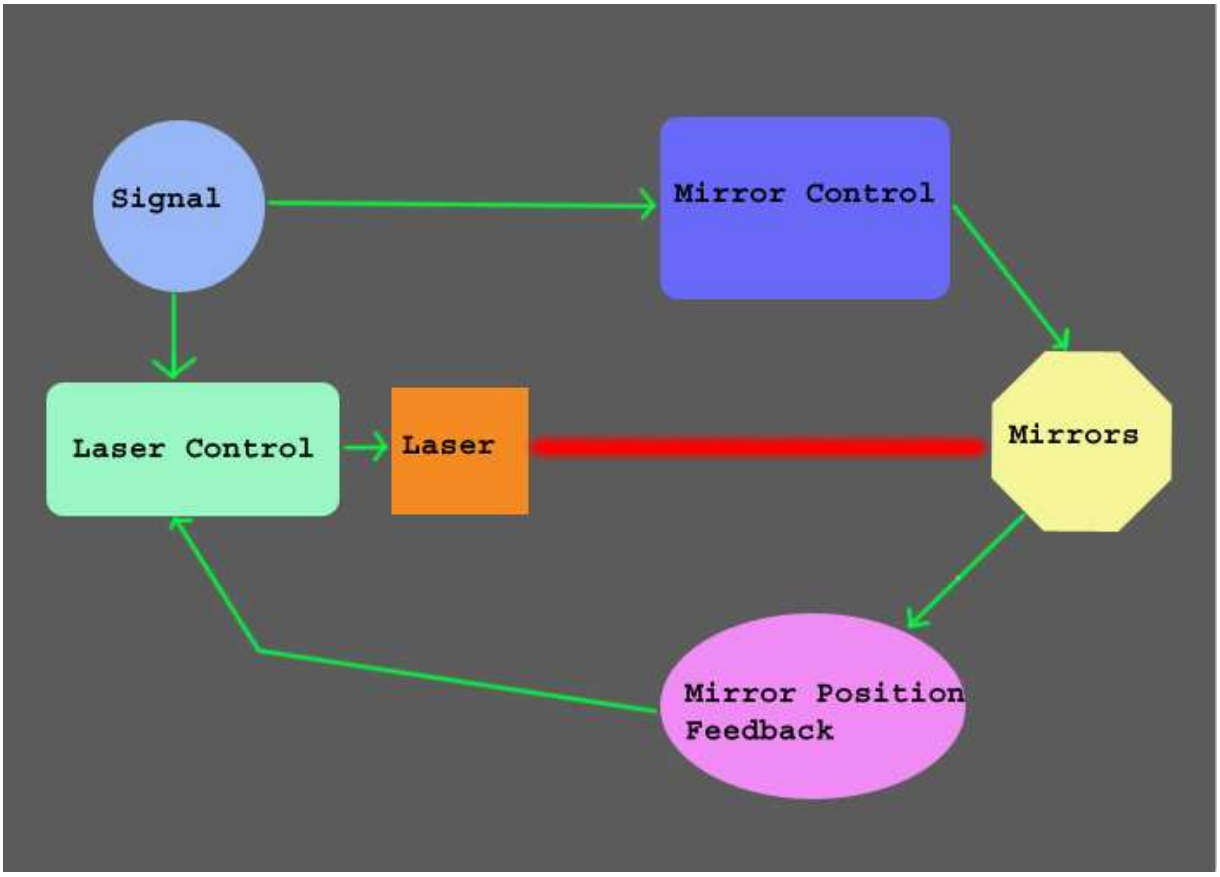

Figure 1: High-level block diagram

1 High-level Design

Conceptually, the design for the projector is very simple. Some kind of video signal comes into the device. Ideally, this will be a common format, such as VGA or NTSC composite. Based on the video signal, the display laser will turn off or on for a given pixel. The mirrors must be oriented such that they position the laser beam to the corresponding pixel in the display plane.

There should be some feedback from the mirrors to the laser control system, so that the laser's output is synchronized with the mirrors' position.

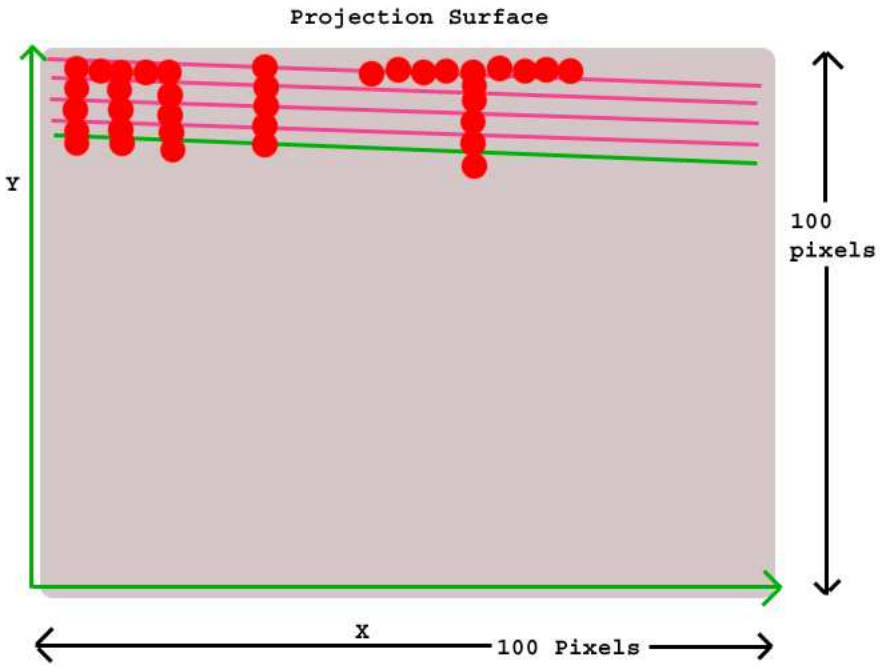

Figure 2: The image projected onto some surface

2 Calculations

In order to design this system, it was necessary to impose some design constraints. The target resolution of the projector was 100 X 100 pixels, with a refresh rate of 10 frames per second.

Initially, all we know is that we want to scan our laser beam across some surface and turn it on at certain points. The projection should look like figure 2:

A sensible system, which we will implement, has two mirrors, one to generate a "line" in one dimension, and the other to spread the line out in a perpendicular direction to generate a planar image.

Based on this, we were able to get an idea of what microcontroller would be necessary to generate the video. It would have to be capable of generating $100 * 100 * 10 = 100000$ pixels per second, so assuming 5 instructions per pixel, it would have to be capable of .5 Million Instructions Per Second (MIPS). However, microcontrollers with this capability were easily obtainable, and it became evident that the limiting factor would be the mirrors and the motors that would move them. In order to achieve the desired resolution, the horizontal scanning mirror would have to scan at $100 * 10 = 1000$ faces per second. Assuming a 10-sided polygonal mirror, the motor would be turning at 100 revolutions per second, or 60,000 revolutions per minute! This was considered unreasonable, and the refresh rate was dropped to 1 frame per second for the preliminary system, lowering the necessary RPMs to 6,000, which is more reasonable.

The vertical mirror could turn at a much slower rate, for while the horizontal mirror would have to project a point for 100 units on the projection surface, the vertical mirror only has to move the point 1 unit. A 600 rpm motor is very easily obtainable.

The other limiting factor was the laser control system. If we assume 1 bit grayscale resolution (black/white images), then the system has to switch the laser $100 * 100 * 1 = 10000$ times per second. However, this is not an unreasonable number for mid-to-high

speed electronics, so it did not impose any additional constraints on the design.

There are very few electrical calculations involved. The only non-trivial one was the variable current source for the laser diode. This was simply an NPN transistor with a TTL level signal at the base switching it, the diode connected from V_{dd} to the collector, and a variable 1 k Ω resistor between the emitter and ground.

3 Construction

The functionality of the projector revolves around good control of the laser, so the first thing to do was to get control of a laser from a microcontroller. Due to limited availability of optical equipment, the easiest way to control the laser was to control a current souce from a microcontroller, and then use the current source to directly drive a laser diode. A simple current source was built from a single transistor, and it had an unloaded response up to 100mHz, much higher than this project requires.

An Atmel atTiny26 microcontroller was chosen for use in this project. It is a 16 mHz 8 bit micro with no external components needed. Most of the coding was done in assembly to ensure proper timing. A simple program was written to flash a 5mW laser diode at different frequencies, demonstrating control of the laser.

The next step was to achieve one-dimensional control of a laser image. A hexagonal scanning mirror from a laser printer with its own speed control hardware was used, and it spun at 8170 rpm. Putting a collimating lens over the laser diode, and using the flashing program on the microcontroller, the spinning mirror created a line. However, because there was nothing synchronizing the microcontroller's flashes to the faces of the mirror, the differently placed "dashes" created by each face of the mirror as it rotated would overlap one another, creating what appeared to be a blurred line.

To remedy the synchronization problem, an infrared emitter/detector pair was aquired. The unit was mounted facing the scanning mirror, such that the mirror would only the reflect emitted light into the detector at a certain angle. Using this, it is possible to get an electrical signal that tells the microcontroller that the hexagonal mirror is on a new face. This can be used to synchronize the laser with the position of the mirror, so as to begin drawing each line at the same point relative to the mirror's rotation.

Figure 3: Line Generation

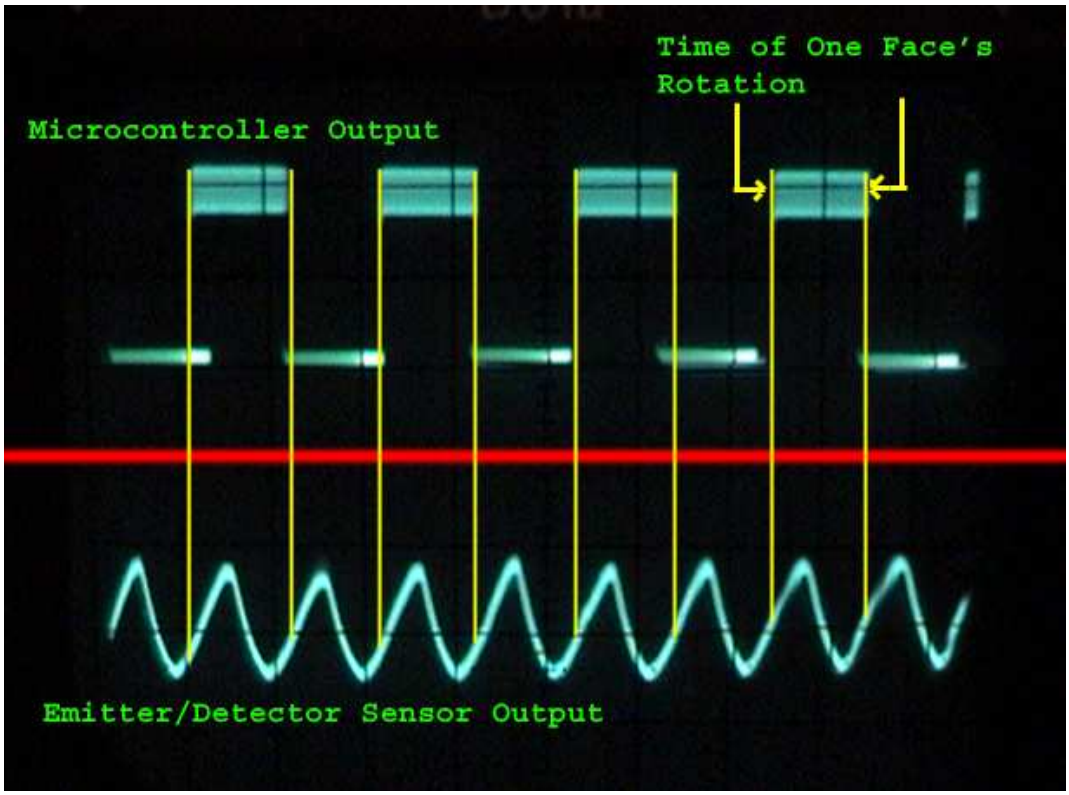

Figure 4: Dual traces of the sensor output and the microcontroller running a rising-edge detection program

4 Construction Continued

The polygonal scanning motor used in this project was from an old laser printer. It is part of a motor module with closed-loop feedback to ensure that the motor rotates at 8170 rpm. Since we know it is moving at a fixed rate, it doesn't matter where we put the emitter/detector sensor, since the time between detections will be constant (as shown in Figure 3), so we can put it anywhere and simply add a delay before we begin drawing our line to ensure that we begin drawing where the face starts. After trial and error determining various timing parameters and the drawing delay, we were able to generate a dash fixed in space. This is the first step towards being able to generate a controlled image. The next step was to draw a dashed line. The target pattern for the final 2D projection is a checkerboard. The dashed line is fairly simple, as it is simply a modification of the program that produced the dot. The programming for the microcontrollers was originally done in C, but the timing requirements for the projector were too tight to be properly met with a compiled language, so the code was rewritten in assembly. The atTiny has an external interrupt, which was configured to propagate through an interrupt vector on a rising transition on an input pin. The interrupt vector simply jumps to a subroutine which performs timing functions at $\frac{1}{clockspeed} = \frac{1}{16mHz} = 6.25X10^{-8}$ second resolution. The subroutine turns the laser on and off for precise intervals, generating "white" spaces and colored pixels.

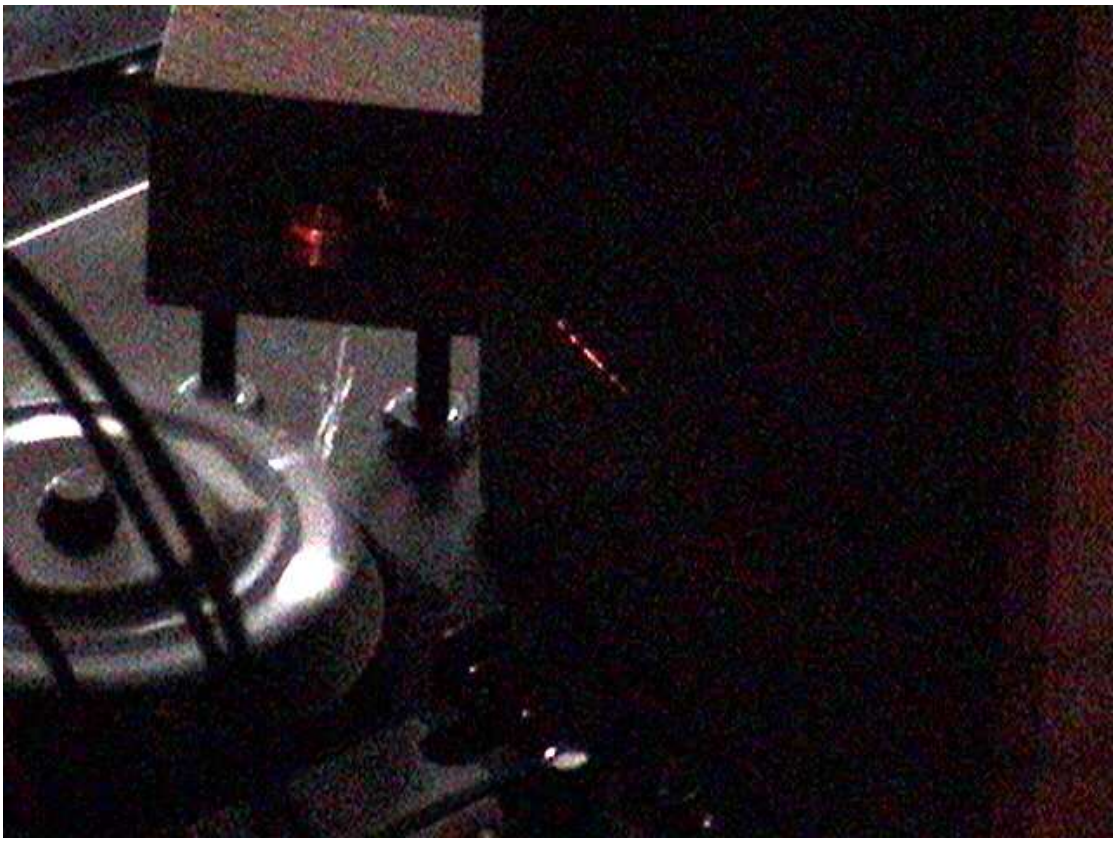

Figure 5: A stationary dash

Figure 6: Generating a dashed line

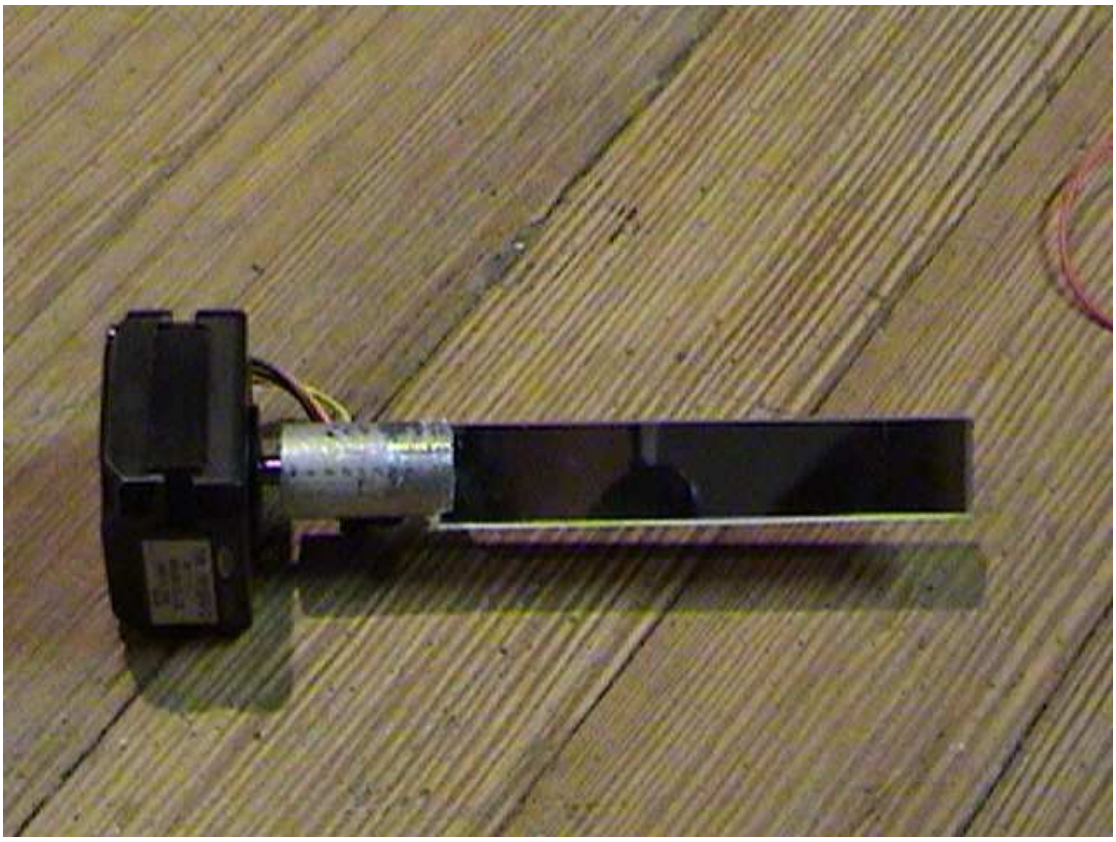

Figure 7: The stepper motor with mirror attached

5 The Second Dimension

Adding the second dimension to the projection is a little more tricky than adding the the first. As the system draws a line in the first dimension (Arbitrarily, we will refer to the line drawn by the polygonal mirror as the X axis, and the line perpendicular to that as the Y axis), another mirror must position that line along the Y axis. The Y mirror must be synchronized to the X scanning mirror. The Y mirror does not need to be as fast as the X–it should be slower by a factor of the number of horizontal pixels–in this case 100. Due to the control issues posed by using another DC motor for the Y, we chose instead to use a bipolar stepper motor with a resolution of 0.9deg per step. A long mirror was cut from rear-coated lexan stock, and an adapter was machined from aluminum to attach the mirror to the stepper motor.

Before the motor was attached, a permanent mount was made for the X mirror, the laser, and the emitter/detector to hold the parts in alignment.

Since the atTiny controlling the laser was already generating an interrupt to synch to the horizontal mirror, and it was many I/O lines left unused, it made sense to use it to control the vertical mirror as well. Four I/O lines were connected to an L293D dual H-bridge chip, which then connected to the motor. By sending out a sequential half-step at each synchronization interrupt, it is theoretically possible to separate each line in the projection by $\frac{r*.45*\pi}{180}$ degrees, where r is the distance from the vertical mirror to the projection surface. However, the stepper motors we aquired were not able to half-step, and so the steppers were ditched altogether.

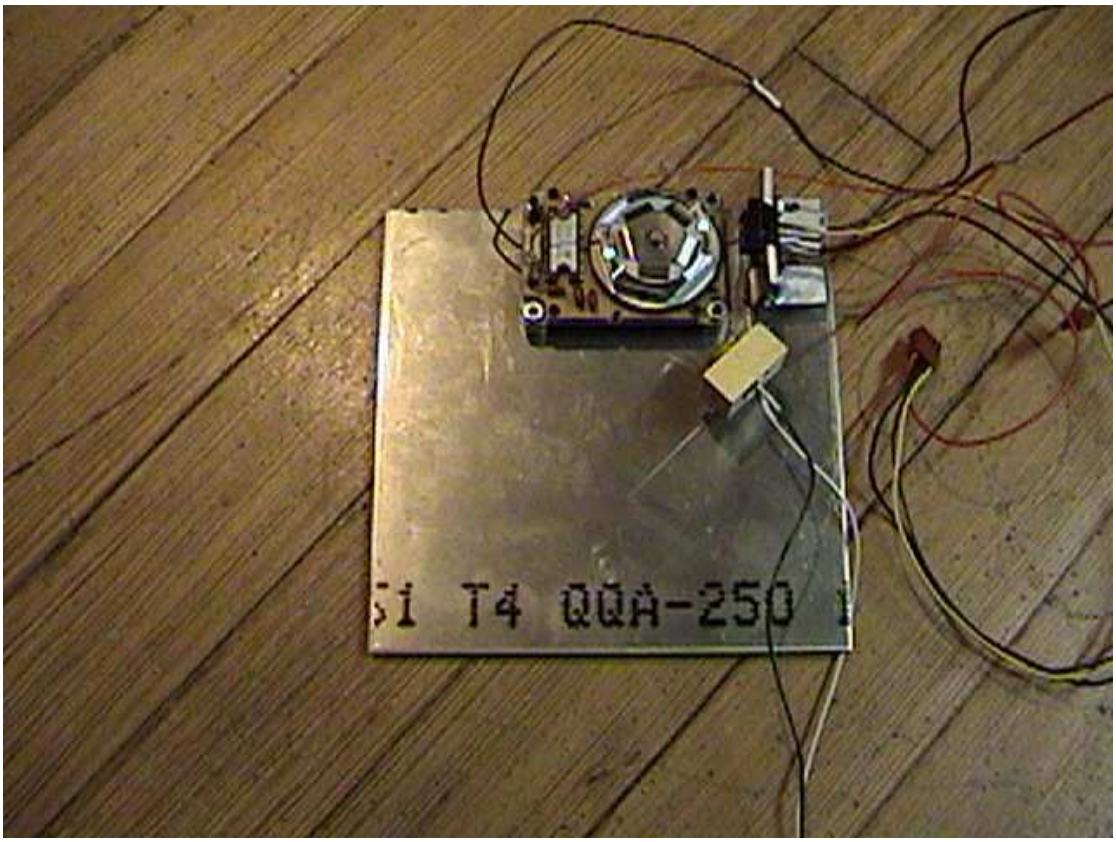

Figure 8: The mounting plate for the laser hardware

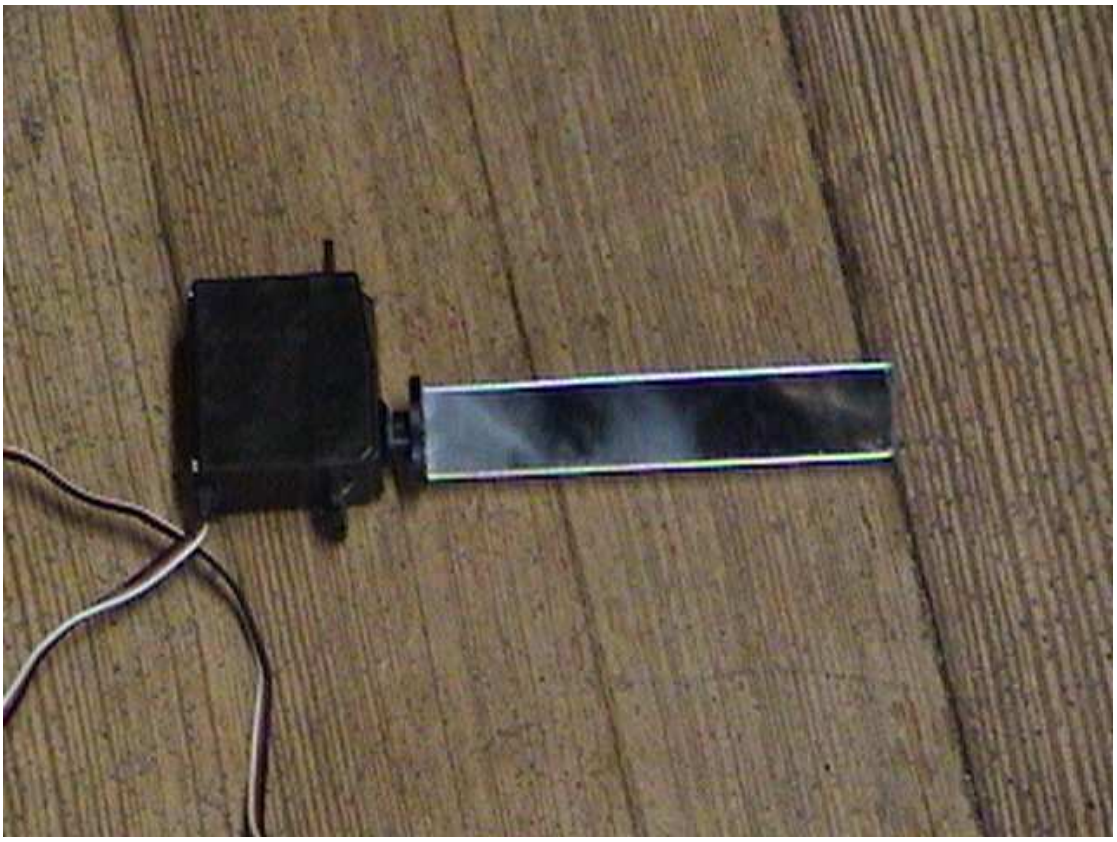

Figure 9: The servo with mirror mounted

Instead, a mirror was mounted on a servo motor. Servos have adequate position control. However, a servo requires a pulse-width encoded signal, and so it became necessary to use a different microcontroller. Since atmels have no built in routines in assembly for specifying a variable-width pulse output, we used a Basic Micro microcontroller which was readily available and had hardware built in for servo control.

The new microcontroller shared a common ground with the atmel board, and it also took the interrupt generated by the emitter/detector as an input, to determine the angle of the servo.

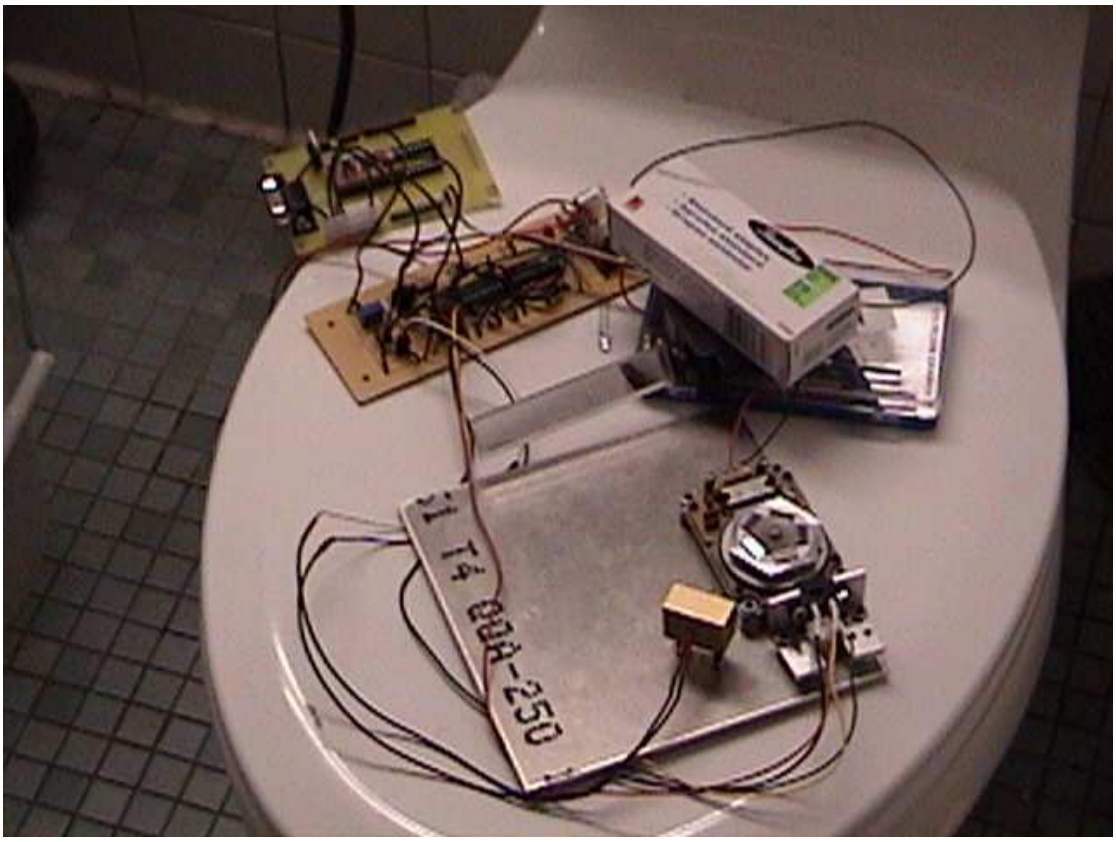

Figure 10: The final version of the electronics and optics. A bench power supply is out of view. The device is on a toilet because the bathroom was the darkest place to observe and attempt to photograph the projection.

6 Results

The final result of this project was a working, two-dimensional projector. Unfortunately, the 5mW laser diode used was too weak to produce an image at an appreciable distance. However, one of the strengths of this device is that the electromechanics are separate from the optics, so a stronger laser can be swapped in with little effort. The resolution was not as high as we had originally aimed, closer to 50 X 50, but the total money spend on materials for the project was under fifty dollars.

The maximum power consumption was 12V at 400mA, or 4.8W, but much of this was dissipated over voltage regulators and status LEDs. With proper attention paid to power conservation, the device could be made to be much more efficient. Also, if smaller motors were used, they would have less inertia, and could spin faster using less power. With proper development and research, this could be made into a projector that could fit into the palm of one's hand and run off of 2 AAA batteries.

7 Conclusions

We are currently at a crossroads in portable electronics technology. We are advanced enough to make devices miniscule, but we cannot because they then cannot be used. There is a high demand for nontraditional user-interfaces, and this is a viable option. There is certainly a lot more work to be done on it before it could be used widely. However, this design is easily extendable to color graphics, and with clever software/hardware tricks such as interlacing, the video resolution and refresh rates could meet current standards. There are several directions to continue with this project. Using PWM to achieve variable brightness monochrome images, adapting the software to display VGA or NTSC input signals, and a robust shockproof design, to name a few. However, what we have demonstrated with this project is that it is possible to create a hi-tech projector using only old parts and very little money. With more work, projectors like this could become commonplace in our lives.

8 Acknowledgements

I would like to thank my advisor for this project, Dr. Jim Bales.
I would also like to thank Mark Tobenkin and Brian Neltner for listening to me gibber about the things I would like to do with this projector.
I would also like to thank Jonathan Barchi and Samuel Xerxes, for their help in attempting to photograph the final projection.

9 Appendix–Atmel Code

```
.include "tn26def.inc"
.def motor = R16
.def temp = R17
.def index = R18
.def del = R19
.def count = R20

.org 0x0000
RJMP RESET
.org INT0addr
RJMP mirror_int


RESET:
ldi index, 0x00
ldi temp, 0b10111111
out DDRB, temp
ldi temp, 0xFF
out DDRA, temp
ldi motor, 0x00

ldi temp, 0b00000000
out PORTB, temp  ;  pullups

; set up int0 and int1

ldi TEMP,0x03 ; interrupt t0 and t1 on rising edge only
out MCUCR,TEMP
ldi TEMP,(1<<INT0) ; int0 mask set
out GIMSK,TEMP

sei ; enable interrupts and off we go!


MAIN:
RJMP MAIN

CONTINUE:
ldi count, 0x00
ldi del, 0x00
WAITSTART:
inc del
cpi del, 0x0A
BREQ ON
RJMP WAITSTART
```

```
LOOP:
inc count
cpi count, 0x05
BREQ MAIN

ON:
ldi motor, 0b01000000
out PORTA, motor
ldi del, 0x00
WAITON:
inc del
cpi del, 0x03
BREQ OFF
RJMP WAITON
OFF:
ldi motor, 0x00
out PORTA, motor
ldi del, 0x00
WAITOFF:
inc del
cpi del, 0x05
BREQ LOOP
RJMP WAITOFF

mirror_int:
sei
RJMP CONTINUE
```