Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

Examining Gyroscopes for 3-D Position Tracking in a Non-Destructive Evaluation System

Rafic Bachnak, Andres Rubio, Jaime Garcia, Sofia Maldonado, Abel Medellin
Systems Engineering
Texas A&M International University
Laredo, Texas, USA
rbachnak@tamiu.edu

***Abstract*—This paper describes the use of gyroscopes to provide three-dimensional data acquisition capabilities to an eddy current inspection system. The pprototype combines positional and eddy-current data to produce a C-scan of tested material. The preliminary system consists of an eddy current probe, a position tracking mechanism, and basic data visualization capability. Test results are presented and briefly discussed.**

***Keywords*--position tracking, gyroscopes, non-destructive evaluation, eddy current, visualization**

I. INTRODUCTION

Nondestructive Evaluation (NDE) refers to the process of locating defects and providing some measurements about the defect such as length, depth, and orientation. NDE plays an important role in ensuring that components and systems are free of defects that compromise their functionality. Non-destructive testing is performed in such a way that objects under inspection are not damaged or affected in any way. While there are several testing methods, the three widely used techniques for materials testing and evaluation are radiography, ultrasonic, and eddy-current. In the case of eddy current, electrical currents are generated in a conductive material by an induced alternating magnetic field. Interruption in the flow of current due to imperfections or changes in the material's properties indicate some type of flaw. The eddy current method has been used in several industries, including space [1, 2] and chemical processing [3, 4]. Some specific applications include detection of cracks [5, 6, 7, 8, 9], measurements of material thickness [10], determining metal thinning due to corrosion [11], measurements of coating thickness [12], conversion coating [13], determining electrical conductivity [14], heat damage detection [3], and detection of corrosion in heat exchanger tubes [4]. This paper presents progress in the development of an eddy current prototype that combines positional and eddy-current data to produce a visual representation of materials being tested [15]. Gyroscopes are used to provide three-dimensional capabilities to the position tracking mechanism of an eddy-current inspection system.

Using data from computer mouse with a 3-D gyroscope attached to it, visual representation of a surface is generated and displayed using AutoCAD. The heart of the system is the velocities program that allows gyroscopes to manipulate the X, Y, and Z axis in AutoCAD. The other part of the system is the interface between hardware and velocities program. The following sections describe the system software and hardware.

The hardware design consisted of two phases. In the first phase, a device that simulates the outputs of a MEMS gyroscope was created and used for testing of the Velocities program. In the second phase, the outputs of a MEMS gyroscope were interfaced to the velocities program. A MEMS gyroscope is an electronic device that measures the rate of change about an axis and outputs an analog voltage. Both trials were accomplished by the use of a microcontroller.

Gyro simulation was accomplished by using the RCTime command of the Basic Stamp 2 (BS2) microcontroller. This command allowed us to detect the position of a potentiometer by correlating the resistance to RCTime Unit. The data outputted from the BS2 corresponds to the time required for the voltage in an RC circuit to drop from 5v to 1.4v. This measurement is scaled by the BS2 and is not given as a measurement of time but as an RCTime Unit. By measuring the resistance of the potentiometer at predetermined positions of 0, 90 and -90 degrees we were able to determine the position of the potentiometer as a function of RCTime Unit. The formula is: Angle = .02382 X RCTime Unit + 118.3854. All computations are done in a C++ program.

To interface a MEMS gyro to the velocities program, the output of the gyros was converted from analog to digital. To do this, an 8-bit A/D converter that gave a resolution of 19 mV was used. This converted output is sent to the velocities program via the BS2. The following sections provide additional details.

II. POSITION TRACKING

Position tracking is accomplished by using a laser computer mouse, MX-1000 by Logitech. The laser mouse was tested using a milling machine and AutoCAD. Average % error varied between 1.5% and 7.6% [15]. It is important to note that the mouse works by shining a light source, in this case a laser, at an angle onto the surface and using an imaging sensor to collect the bounced light. Using motion detection algorithms, relative motion of the mouse is determined. However, this motion is relative to the sensor chip. If the sensor chip is rotated, then the relative motion vectors also rotate. With a single mouse there is no way to detect this rotation.

A. Development of a Prototype

Fig. 1 shows the hardware used in the prototype. The major components are an eddy current system, the Nortec 2000D+, an eddy current probe, a Copperhead laser mouse by Razer, and a data acquisition device, USB-1208FS. The USB-1208FS interfaces the Nortec 2000D+ with the laptop.

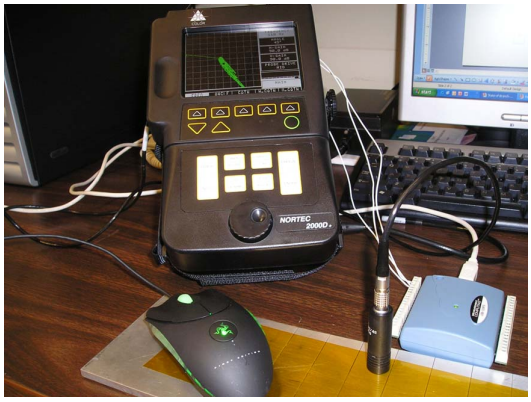

Figure 1. Prototype major components.

B. Acquiring Positional Data

To be able to produce a visualization of defects, positional data is combined with eddy current data to produce a C-scan of the examined material. The MX-1000 mouse has a 800DPI resolution and an update rate of 250Hz. To be able to detect a feature that is .050", then a sample is needed every .025". That means the probe can travel no faster than .025"*250/s = 6.25"/s. To scan the target surface of 12"x12" it would take about 1.92s for each scan. To detect features .050", a scan every, .025" would be required, or 480 scans, which would take a minimum of 15 ½ minutes. The current system uses a cross-platform API, called ManyMouse, that abstracts using multiple mice into a single small library.

C. Digitizing Data from the Probe

The Nortec 2000D+ eddy current system, the device used in the prototype, operates at 6000 Hz. Along with displaying the sensor information on its built-in screen, the Nortec 2000D+ also has two analog outputs that correspond to the horizontal and vertical positions of the cursor on the screen. These outputs must be digitized before they can be used [16]. For testing purposes, a Measurement Computing USB-1208FS digitizer was used. This 12-bit digitizer has a USB interface which allows easy interface with the laptop. It also allows the display of information as it is captured on the laptop's screen.

The USB-1208FS digitizer is not designed to be a low-latency digitizer. That is, it cannot digitize samples at 6000 Hz. Since two signals have to be sampled, its effective capture rate is halved. However, since the positional data is only updated at 1000 Hz, the digitizer only needs a comparable speed. The USB-1208FS is giving around 150 Hz, which is adequate for testing purposes. The USB-1208FS can be put in continuous scan mode, or in single capture mode. In continuous scan mode, it captures multiple samples and returns them. In this mode it is capable of capturing at much higher rates, on the order of 100 KHz. However, in order to use the mode, a complex multithreading system that has a high-precision clock would be required. It is also possible to use a more expensive and faster digitizer.

The software library that comes with the USB-1208FS is binary-only, and only works on Windows XP. Currently, a Linux driver, written by Warren Jasper of NC State, is used [16]. This is the only piece of the system that is not platform independent. However, it is only an initialization routine, and a call to collect a sample that needs to be updated.

D. Testing the Prototype

The prototype was tested by using the mouse in conjunction with the eddy current probe to create a visualization of the surface flaws on an aluminum plate. The plate is approximately .250" thick and has 8 grooves 1" apart. The grooves are rounded at the bottom and are .015" thick. The depths of the grooves are: .005", .010", .040", .060", .080", .100", .100", and .200". Since the grooves are rounded, the first groove isn't as thick as the others, but about .010" thick. Protective tape was applied to the plate to keep it from getting scratched by the probes. The probe and mouse were held together as shown in Fig. 2. Fig. 3 shows a visualization of the front side of this test plate using a pencil probe operating at 300 KHz. The visualization was made with our system that determined position using a single Copperhead mouse and a USB-1208FS digitizer. Since the mouse and probe were not held a fixed distance from each other, and since the mouse was held by hand there are positional errors.

Visualizations were developed as C++ programs written for Linux. They use OpenGL [17] for performing the graphics. OpenGL is a cross-platform graphics library. All of the code for this project, except for the Linux-only USB-1208FS Linux driver, uses cross-platform libraries that have source code available. Even though C++ is used, there are few advanced features used, and with a small effort the entire project can be recompiled in C.

The prototype was also tested for a backside inspection of the plate. The backside inspection was completed with a 100 Hz – 20 KHz frequency probe. Since the probe is much thicker then a pencil probe (0.6"), it responds to the same crack over a larger distance. This causes much larger lines in the visualization.

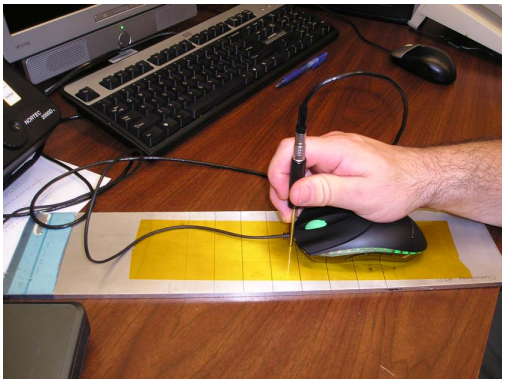

Figure 2. Scanning an aluminum plate.

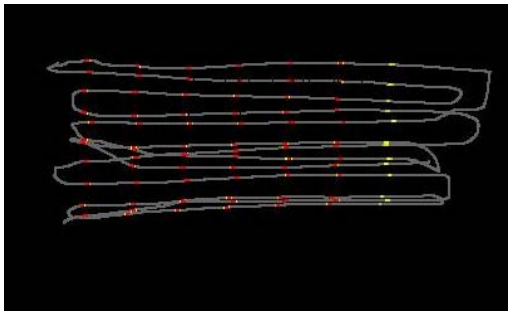

Figure 3. Visualization of plate.

III. GYRO SIMULATION

The purpose of gyro simulation was to provide the Angular Velocities program with gyro like inputs that vary at real time and to identify any conflicts prior to the purchase of MEMS gyros.

A. The Software

1) Function: The purpose of the program is to acquire 3-D data using gyroscopes mounted on a computer mouse, while scanning a surface. Sample data is shown Fig. 4. The idea is to have the mouse maintain its normal 2D function while the gyroscopes provide the angular position of the mouse in relation to the starting point, or origin. The main program works by reading "vectors" created after every predetermined delay. For example, if the time precision in the program is set to 1 ms, then the program will read the distance traveled by the mouse in X and Y after every 1 ms.

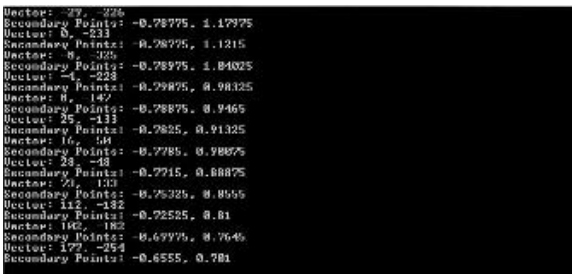

Figure 4. Command Prompt receiving and saving points at 1 kHz.

The default value for this feature is 1 ms since it's the smallest delay that the program can produce effectively. These vectors are recorded and added to form what appear as a free hand sketch of the mouse trajectory, as plotted in Fig. 5.

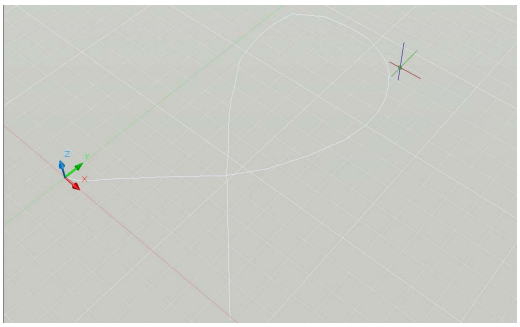

Figure 5. About 1.5 foot trajectory sketched with 1270 lines.

A couple of seconds after the user stops moving the mouse, the program will stop and save all the coordinates in a script with a layout "readable" for AutoCAD. These can be examined by the user directly or executed in AutoCAD as a script file and thus a lot easier to read. Judging visually, the program works extremely well. The mouse was tested informally for the first time by signing a name with the mouse on the entire desk. The name came out very well, with hardly any rough edges at 100 Hz. The program can now run as fast as 1 kHz but can be slowed down to save memory and CPU power.

The program also processes angular velocities. The rotation program works by scanning the information sent to the USC.scr file by the Pbasic program that was collected from the potentiometers. This information is written in binary numbers since the potentiometers send changes in voltages. There are three potentiometers with each having a specific axis, for example, x, y, and z. The scan is a loop that constantly takes in letters and digits. An example is, X= 12, Y=24, Z=56. All of this information is not formatted to manipulate AutoCAD. The rotation program only gets the numbers obtained in each loop and uses a formula to transform them into degrees per second, which can now be sent to the main program. It sends the "velocities" to the main program where they are organized and placed as AutoCAD commands in between the "line" commands in the original script. The main program also detects the change of degrees by subtracting the first angular velocity by the second and so on until the loop is stopped. When executing the script file in AutoCAD, the "line" and "UCS" commands alternate refreshing both the distance and 3D direction of each new line or vector

2) Additional Functions: A main menu is displayed at the beginning of the program where the user is able to change some settings (See Fig. 6).

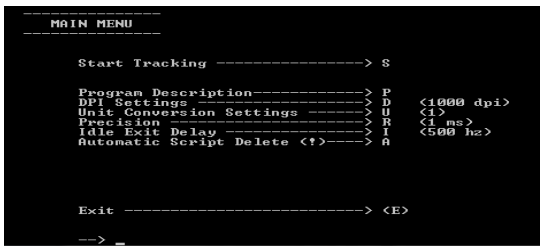

Figure 6. Position Tracking Main Menu.

DPI Settings can be changed in case multiple mice with different DPI are used. If the DPI on the mouse is lowered to a more stable 2000 DPI, the program can be changed accordingly. The default measurement unit in the program is the inch, but this can be modified by changing the Unit Conversion Multiplier. If the user wishes to change the units to feet, all he/she has to do is select "U" from the menu and input a "12". Likewise if the user wishes to change the unit to centimeters, the user can input "0.3937008". The default time delay between each point is 1 ms. To end the sketching process, the user has to let the mouse idle for a couple of seconds. The Idle Exit Delay allows the user to customize how

long it will take before the program ends the sketching process. Fig. 7 shows the end of the trial.

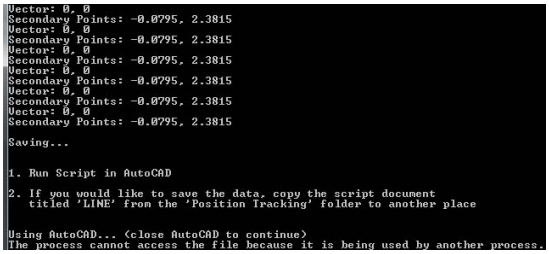

Figure 7. End of tracking trial.

3) Work-in-Progress: The most important thing that must be done next is identifying the precision that will give a 1:1 ratio from dots-per-inch to pixels. Windows has the option to change the "Pointer Speed" of the mouse. This multiplies the DPI by a certain number to travel more pixels. This means that the user must be checking this option every time he/she wishes to use the program. As of now we have not been able to identify the combination of velocities that will give a 1:1 ratio. This is hindering our ability to start testing this portion of the project. A program that allows us to calibrate the mouse is needed. We would like to have exact 1:1 dpi to pixel ratio before we start testing. A code was created to send information via serial port. This code will allow the program to get information in real time and become more precise.

Another part of the program that needs work is the Automatic Script Delete. We already found the C++ commands that will delete everything in a file; all we have to do is implement them. A sub-menu is also planned in the "Unit Conversion" option so that changing to centimeters, inches, feet, meters, and other common measuring units is easier.

Testing of the mouse has led us to the conclusion that it is very difficult to place the optical sensor right above a desired point. A way to solve this would be to place a cross-hair, like the ones used on digitizers, right in front of the mouse. The distance between the cross-hair and the sensor will be calculated in dpi, and taken into consideration when plotting.

4) Testing: A couple of tests have been conducted. It's worth noting that these tests were conducted manually without the use of an accurate machine. The measurements were done with a ruler as seen in Fig. 8.

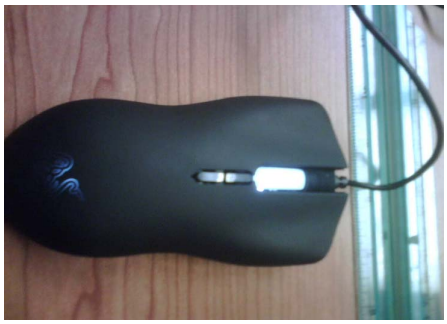

Figure 8. Testing Mouse with ruler.

The purpose of this experiment is to find out whether the scaling of the mouse is somewhat accurate instead of checking pixel-size errors. The data obtained is shown in Table I and Table II.

TABLE I. MOVING MOUSE LEFT TO RIGHT 1 FT.

Trials	Actual Length	Measured Lengthwith Mouse
Trial 1	12 in	12.3615 in
Trial 2	12 in	12.6762 in
Trial 3	12 in	12.4650 in
Trial 4	12 in	12.4628 in

The mouse itself is not making any errors that we know of. The problem is the scaling due to the sensitivity meters on the configuration program. After the previous test, the sensitivity of the mouse was readjusted and a new test was done.

TABLE II. MOVING MOUSE LEFT TO RIGHT 1 FT.

Trials	Actual Length	Measured Length with mouse
Trial 1	12 in	12.0238
Trial 2	12 in	12.3058
Trial 3	12 in	12.5083
Trial 4	12 in	12.1608
Trial 5	12 in	12.5135
Trial 6	12 in	12.0570

The scaling cannot be adjusted well at this point since we don't have a way to precisely move the mouse along a straight line. Nonetheless, it can be concluded that the results were acceptable.

B. The Hardware

Gyro simulation is accomplished by an RC circuit and a microcontroller. The RC circuit is created with a fixed capacitor and a variable resistor (potentiometer). The potentiometer is fixed to a platform with 0, 90, -90 degrees marked. Varying the potentiometer produces changes in the RC Time constant ($\tau = R \bullet C$). This change is detected by the microcontroller, which measures the time required for the capacitor to drop from 5 to 1.4 volts, or the time required to go from logic high to a logic low. The time a capacitor takes to drop from some initial voltage to a final is a function of its RC time constant and is described by $t = \tau$. The microcontroller scales the time by 2µs and outputs a value. The value is then inputted to the angular velocities program as a RC Time Unit. Two trials were conducted to determine the angular position as a function of resistance (trial 1) and as a function of RC time unit (trial 2).

1) Trial 1: The resistance of the potentiometer is measured at 0, 90, -90 degrees. The data (shown in Table III) was then plotted on the graph below (shown in Fig. 9) to derive an equation of the line.

2) Trial 2: The RC time unit was recorded at 0, 90, -90 degrees and the data (shown in Table IV) were plotted on the graph below (shown in Fig. 10). This method is preferred because it requires fewer computations since the microcontroller outputs RC time unit.

TABLE III. TRIAL 1 DATA

Angle	Resistance (Ω)
-90	9180
0	4970
90	1660

Angle = -3.52896 x Resistance + 105.8688

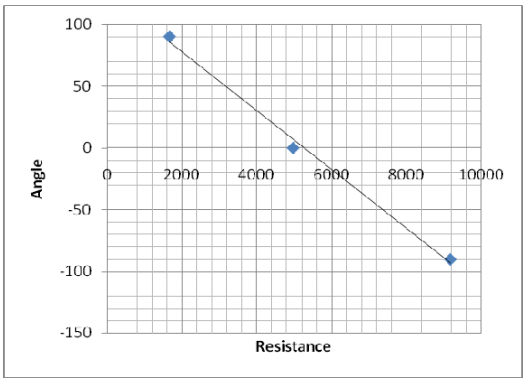

Figure 9. Trial 1 results.

TABLE IV. TRIAL 2 DATA

Angle	RC time Unit
-90	56
0	30
90	5

Angle = -0.02382 x RC time Unit + 118.3854

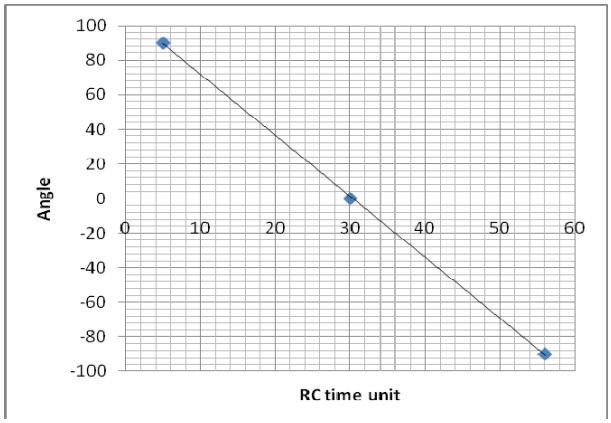

Figure 10. Trial 2 results.

C. Microcontroller

The Basic Stamp (BS2) command used to produce a varying output is RCTIME. The BS outputs a RCTIME Unit as described earlier. By using a potentiometer to vary the resistance in RC circuit, we can manipulate the RCTIME Unit and subsequently measure the change in position of the potentiometer. In the basic stamp program a delay is inserted to allow the capacitor to charge to 5 volts then the discharge time is measured. This value is then outputted via a serial connection to the PC. HyperTerminal is used to monitor the serial port. The HyperTerminal is in capture mode to send the data to the notepad file and will later be extracted by the Angular Velocities program. The data is in the following format.

X = ? (? = RCTIME Unit)
Y = ?
Z = ?

This value is then inputted to the formula derived in trial 2. The result is angular position. Angular velocity can be derived from angular position by finding the rate of change.

Angular velocity = $(Angle_n – Angle_{n-1})/time$ = degree/ms
$Angle_n$ = Current Angle
$Angle_{n-1}$ = Previous Angle

The computations are done using C^{++} because the angular velocity can be negative in direction and negative numbers not supported by the PBasic programming language.

D. RC Circuits

Three identical RC circuits are used to detect a change in position. These circuits use R1and C1, R2 and C2, and R3 and C3, as shown in Fig. 11. The resistors R1, R2, and R3 are in parallel with its respective capacitor. There are three 220Ω (R5-R7) resistors placed in series with the RC circuits. This protects the BS2 microcontroller in the event that the potentiometer is tuned to 0Ω. The code to perform this function is:

```
HIGH 7              'Puts a High on pin 7 to charge cap
PAUSE 10            '10ms to charge cap completely
RCTIME 7, 1, X      'time it takes volts to drop from 5 to 1.4v
```

1) Data Output: The position data is transferred to the computer via a serial port. The position data is stored in a notepad file until it is extracted by the Velocities program. The serial output command is:

```
SEROUT 16, 16468, [DEC ? X]   'Data sent to PC. 9600 baud, 8 bit no parity and decimal
SEROUT 16, 16468, [11]   'Ascii 11 = return; create a new line in notepad
```

2) Power Circuit: A 12v 100mA power supply is used to supply power to the circuits. The KIA 7805 P regulates the voltage to 5 volts.

IV. GYRO INTERFACE

Several circuits were designed in preparation for the implementation of gyroscopes. There are two major sections of these new circuits: analog to digital conversion and a summing op amp circuit.

An analog to digital converter (ADC) is used to convert the gyros analog output voltage. An ADC 0831 C is used for this purpose. This converter is an 8 bit differential converter and is limited to only convert positive voltages. The circuit is currently setup to convert from 0-6 volts. This can be changed to accommodate the gyroscope output by changing the reference voltage to the A/D converter. Eight bits of data will produce a resolution of .0235294 volts with a reference of six

volts. There is currently some systematic errors induced by the analog to digital conversion process and as shown in Table V.

TABLE V. ANALOG TO DIGITAL CONVERSION

BCD	Voltage Computed (V)	Voltage Actual (V)	Error
255	5.999997	6	.000003
128	3.0117632	3.0117642	.0000010
64	1.5058816	1.5058818	.0000002
32	.7529408	.7529411	.0000003

Voltage Computed = BCD x .0235294
Voltage Actual = 6BCD/255
Error = Voltage Actual – Voltage Computed

The BS2 microcontroller is used to set up timing, converts the digital signal from the ADC to BCD and to output this data to the Angular Velocities program.

The ADXRS 401 requires a 2.5 and 5V input to provide power. Currently, two power supplies are being used. The gyroscopes should output a null voltage of 2.5V and an increment of .015V degree/second. Currently, the gyros null voltages are different for each one. There is a null voltage adjust circuit that can be implemented to correct this and can be found in the gyro data sheet.

The gyroscope outputs a positive voltage for clockwise rotation and negative voltage for counterclockwise rotation. The orientation of the gyroscope needs to be determined and should be accounted for in the Angular Velocities program.

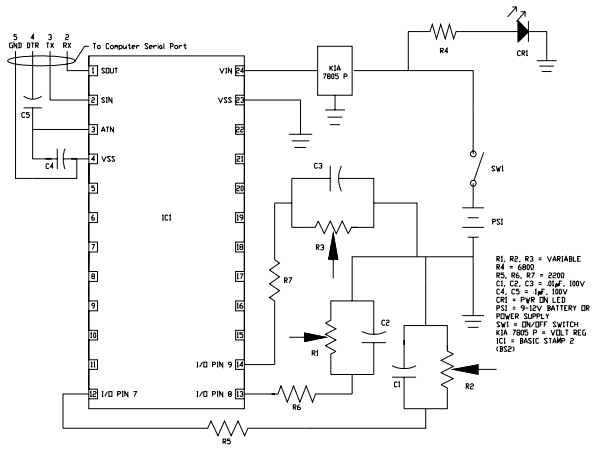

Figure 11. Circuit diagram.

V. CONCLUSION

This paper describes the use of gyroscopes to provide three-dimensional information to an eddy current prototype that combines positional and eddy-current data to produce a C-scan of tested material. Initial test results of the gyroscopes are not very accurate. The use of an 8-bit ADC is contributing to this. A 12bit ADC will be used to correct this problem. Another known problem is the paring of the XY data with the gyro data. This can be addressed by inputting the gyro data directly into the Angular Velocities program.

ACKNOWLEDGEMENT

This project is partially supported by a grant from NASA-Johnson Space Center and award # P031C080083 from the US Department of Education.

REFERENCES

[1] R. Smith, G. Hugo, “Transient Eddy-current NDE for Aging Aircraft – Capabilities and Limitations,” Proc 4th Joint DoID/FAA/NASA Conf on Aging Aircraft, May 2000.

[2] A. Ahmed, J. G. Bakuckas, Jr., C. A. Bigelow, P. W. Tan, J. Awerbuch, A. Lau, and T. Tan, “Initiation and Distribution of Multiple-Site Damage (MSD) in a Fuselage Lap Joint Curved Panel,” Proceedings of the 5th Joint NASA/FAA/DoD Conference on Aging Aircraft, September 10-13, 2001, Kissimmee, FL.

[3] A. Birring and G. Marshall, "Eddy Current Testing in the Petrochemical Industry," Materials Evaluation, November 2003.

[4] Helle H. Rasmussen, Hans Kristensen, and Leif Jeppesen, “NDT and Heat Exchanger Tubes,” NDT.net October 1998, Vol.3 No.10.

[5] Y. Wu, C. Hsiao, “Reliability Assessment of Automated Eddy Current System for Turbine Blade,” NDT.net, Vol. 7, No. 11, November 2002.

[6] S. Burke, “Crack Depth Measurement using Eddy-Current NDE,” 10th Asia-Pacific Conference on Non-Destructive Testing , 17-21 September 2001, Brisbane, Australia.

[7] D. Moore and F. Spencer, “ Interlayer Crack Detection Results Using Sliding Probe Eddy Current Procedures,” 10th Asia-Pacific Conference on Non-Destructive Testing , 17-21 September 2001, Brisbane, Australia.

[8] B. Wincheski, J. P. Fulton, S. Nath, M. Namkung, and J. W. Simpson, “Self-Nulling Eddy Current Probe for Surface and Subsurface Flaw Detection,” in Materials Evaluation, Vol. 52/Number 1 (January 1994).

[9] Ryszard Sikora, Piotr Baniukiewicz, “The Complete Measurement System for Crack,” Proceedings of the VIth International Workshop, Computational Problems of Electrical Engineering, Institute of Theory of Electrical Engineering, Zakopane, Poland, September 1-4, 2004

[10] T.K. O’Brien and D.C. Kunerth, “Pulsed Eddy Current Thickness Measurements of Transuranic Waste Containers,” US Dept of Energy, Report number INEL--95/00313; CONF-951091-12, 1995.

[11] J. Skramstad, R. Smith and David Harrison, “Enhanced detection of deep corrosion using transient eddy currents,” Proc. 7th Joint DoD/FAA/NASA Conference on Aging Aircraft, New Orleans, Sept 2003.

[12] “Measuring Coating Thickness,” CMI International Inc., Elk Grove Village, Illinois, http://www.pfonline.com/articles/pfd0027.html, accessed on November 7, 2005.

[13] A. Salem, T. Nayfeh, and C Fox, “Nondestructive Quality Assurance Testing of Chemically Processed Wood,” Int. Journal Advanced Manufacturing Technology, Springer-Verlag London Limited, Vol. 26, pp. 1275-1283, January 2005.

[14] Yaron Danon, Changqing Lee, Chris Mulligan, and Greg Vigilante, “Characterizing Tantalum Sputtered Coatings on Steel by Using Eddy Currents,” IEEE TRANSACTIONS ON MAGNETICS, VOL. 40, NO. 4, JULY 2004, pp. 1826-1832.

[15] R. Bachnak and S. King, “Non-Destructive Evaluation and Flaw Visualization Using an Eddy Current Probe,” The Third Int. Conf. on Systems (ICONS 2008), CD-ROM Proceedings, ISBN: 978-0-7695-3105-2, pp.134-139, Cancun, Mexico, April 13-18, 2008.

[16] Manymouse, http://www.icculus.org/manymouse/, visited July, 2006.

[17] Operating Manual, MIZ-21B Eddy Current System, Zetec, inc., 2003.