Chaotic Rotor

Overview of the Instrument

The Chaotic Rotor apparatus allows quantitative analysis of the motion of a rotating magnetic dipole in an oscillating magnetic field. This motion is analogous to the motion of the driven damped pendulum. Nearly every parameter on the apparatus is adjustable, including the strength of the fixed field, the frequency and amplitude of the drive field, the damping parameter, and the rotational inertia of the rotor. Precise time and position information is reported to a computer via USB at intervals spaced evenly throughout the drive cycle, allowing the experimenter to generate not only phase-space plots of the motion but also a series of Poincaré sections for the motion.

The instrument is controlled via a Cortex-M4 microprocessor (on an Arduino-compatible "Teensy 3.1" development board) which manages all communications with the computer, generates the oscillating drive field, tracks time and position of the rotor, and allows complete user control over the behavior of the apparatus via GPIB-style commands.[1]

Powering the Instrument

There are two sets of Helmholtz coils affecting the rotor. They are powered by two separate DC power supplies: an adjustable DC supply of up to 12V, and a bipolar ± 12VDC supply. 3.3V logic-level voltage is supplied by a 5V-3.3V converter on the Teensy 3.1 board, powered by the USB connection to the computer.

The larger of the two coil pairs generates a fixed magnetic field, which establishes an equilibrium position for the rotor. This field is analogous to the gravitational field for the driven damped pendulum; the advantage with this apparatus is that by changing the current in these field coils one can effectively adjust the strength of "gravity". Field strength is set by adjusting the voltage on an external supply connected directly to the coils. Please adjust the current limit on this supply to no more than 1.0 A.[2]

The smaller inner coils (drive coils) are controlled by the microprocessor

[1] The instrument is not GPIB/IEEE 488.2-compliant as of the current firmware version, but the commands will be quite familiar in format and usage to anyone familiar with GPIB or SCPI communications.

[2] The maximum safe current for the field coil is still uncertain on this prototype unit, but 1.0 A is more than sufficient for creating a fixed field component. If you wish for experimental reasons to go higher, please watch the temperature of the field coils and turn the current down if they start getting warm!

and the associated amplifier circuit on the controller board. The amplifier circuit requires ±12VDC from a supply capable of driving up to 1.0 A. The DC offset of the drive coils is adjusted via a trimpot on the controller board: to adjust it set the output amplitude to zero, set Coil On to true, and adjust this trimmer until the coil voltage at the controller board output is zero.

Communicating with the Instrument

Communications between the Chaotic Rotor apparatus and the computer are via a virtual serial port on USB at a bauad rate of 115,200. Parameters are controlled (or polled) using 4-character GPIB-style commands sent from any serial-capable program. (LabVIEW is an obvious choice, but Python is also quite good for this and one can even control it by typing commands directly into a terminal emulator program.) Commands are case- and whitespace-insensitive.

HELP Return a summary of available commands.

***IDN** Return a description of the apparatus and the firmware version.

***RST** Reset instrument to default conditions: Coil off, Report off, f = 1.0 Hz, A = mid-range, phase and position both set to 0.

***TST** Test for proper centering adjustment. This test will require the next generation of hardware: as of the current version it will invariably tell you that everything is ok by returning 0.

***ESR** Report (and clear) the Error Status Register. Current firmware only sets ESR bit 5 (communications error) so this command will always return either 32 or 0 until we find a way of identifying other errors. Any error will turn on the Error light on the controller board: *ESR turns it off.

FREQ ? | (float) If followed by '?', the frequency command returns the current frequency setting for the function generator. If followed by a floating-point number, it sets the frequency to approximately that value in Hz. The precision of the internal function generator is limited by the method of generating the output waveform. The microcontroller adjusts the output 256 times per drive cycle. It calculates from the frequency the number of microseconds to wait between each step; at higher frequencies the quantization of this microsecond count seriously messes with the output frequency and above 3,906 Hz the output

breaks entirely. For this apparatus, though, 10 Hz is a "ridiculously high" frequency so it's not a problem.

AMPL ? | (0-1000) The amplitude is set on a scale from 0 to 1000, where 1000 is close to an amplitude of 10 V_p. A query '?' will return the current amplitude, an integer value will set the amplitude. Values less than 0 or greater than 1000 will set the amplitude to 0 or to 1000, respectively.

TRAK ? | (1 | 0) By default, the instrument tracks the position of the rotor. The user can turn tracking on or off with 'TRAK 1' or 'TRAK 0', respectively. 'TRAK ?' returns the current tracking setting.

REPT ? | (1 | 0) setting report to 1 turns on data reporting, 0 turns it off. '?' returns the current report setting. Default value is 0.

COIL ? | (1 | 0) Setting coil to 1 turns on the coil drive output, 0 turns it off. '?' returns the current coil drive output state. Default value is 0.

ZERO Defines current position as 0 degrees.

POSN? Returns current position, in degrees.

TIME? Returns current time, in microseconds.

SAVE (0-9) Saves current frequency and amplitude to non-volatile memory in memory slot 0-9.

LOAD (0-9) Sets frequency and amplitude to the values previously saved in memory slot 0-9.

As noted before, this instrument is not fully IEEE 488.2 compatible. One thing that does not work yet is the ability to string commands together with semicolons: "FREQ 2.0 ; AMPL 750" correctly sets the frequency to 2.0 Hz but then sets ESR bit 5 without changing the amplitude to 750. Another issue is that the command parser *always* takes the first four characters of the command. For a GPIB-compatable instrument the commands 'FREQ' and 'Frequency' should work equally well, but the latter will cause an error on this device.[3]

[3] "It's on my list!" –Shreck

Other Adjustments

The damping parameter is adjusted by using the micrometer screw at the top of the instrument to bring the magnet near the rotor flywheel. Useful values of damping seem to occur for distances of 5 mm or less, although this is an area of active research still.

The inertia of the rotor can be changed by physically changing the flywheel. The default flywheel is a fairly lightweight aluminum disk, but there is a much heavier brass disk available also and of course you can use the lathe or 3D printer to make something different. It might be interesting to make an attachment to fit just below the aluminum flywheel that would allow controllable fine adjustment of the rotational inertia. This would make a good project to check off the "3D print a part" lab requirement...

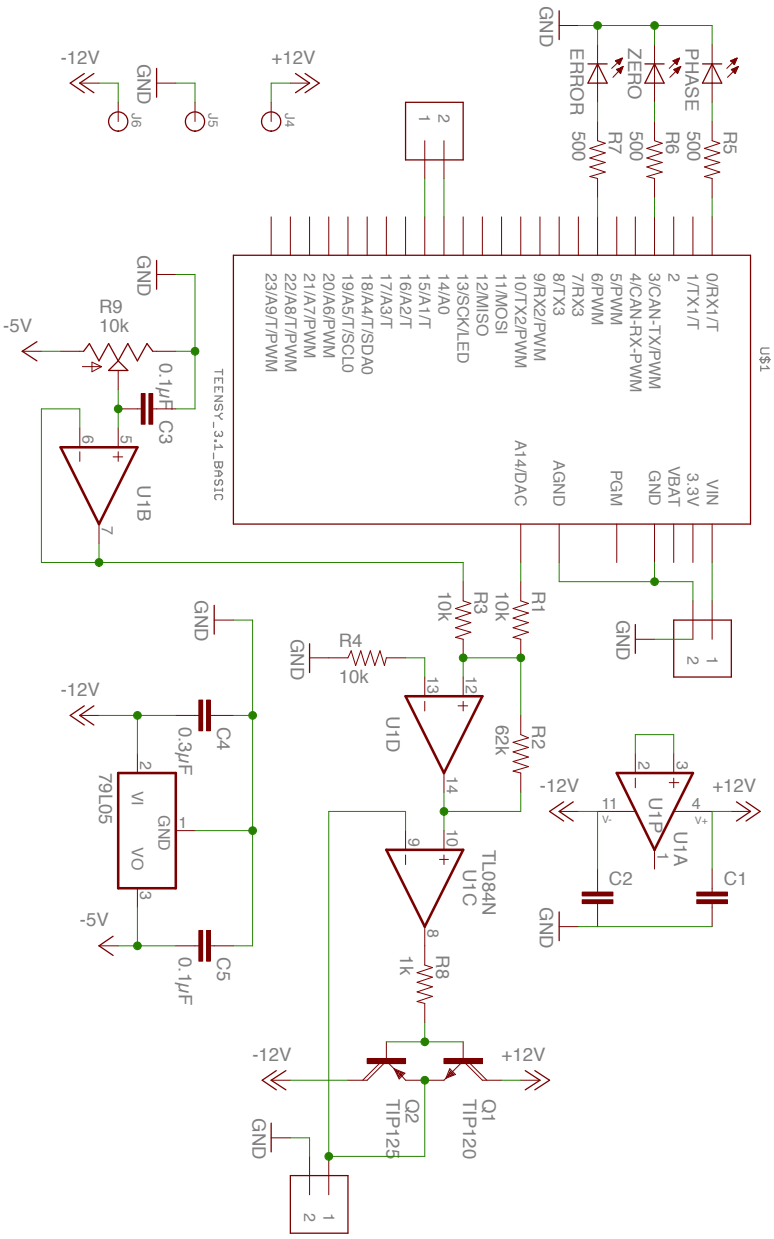
Experiment

This apparatus is new, and those of you using it in Advanced Lab this fall (2015) are the first to actually get data from it! That is of course very exciting, but the downside is that nobody knows yet what constitutes reasonable results. Here are some things that you can try, in order of increasing impressiveness.

1. Write a LabVIEW or Python program to control and collect data from the device. (This is the bare minimum, really.)
2. Find an "interesting" frequency-amplitude area and plot phase-space diagrams for the rotor.
3. Plot a Poincaré section for the rotor, showing the strange attractor.
4. Make a movie showing the evolution of the strange attractor through the drive cycle
5. Compare your results with a numerical solution of the equations of motion for the apparatus.
6. Make a bifurcation map for a given frequency (amplitude) showing period-doubling as amplitude (frequency) is swept.

Version

Experiment manual version 0.5, by Eric Ayars. Written to accompany firmware version Teensy3.1_20150827.


GND
PHASE 500
ZERO 500
ERROR 500
R5
R6
R7
-12V
+12V
J6
J5
J4
U$1
TEENSY_3.1_BASIC
0/RX1/T
1/TX1/T
2
3/CAN-TX/PWM
4/CAN-RX-PWM
5/PWM
6/PWM
7/RX3
8/TX3
9/RX2/PWM
10/TX2/PWM
11/MOSI
12/MISO
13/SCK/LED
14/A0
15/A1/T
16/A2/T
17/A3/T
18/A4/T/SDA0
19/A5/T/SCL0
20/A6/PWM
21/A7/PWM
22/A8/T/PWM
23/A9/T/PWM
A14/DAC
AGND
PGM
GND
VBAT
3.3V
VIN
R9
10k
-5V
0.1µF
C3
U1B
R3
10k
R1
10k
R4
10k
U1D
R2
62k
U1P
U1A
C2
C1
TL084N
U1C
R8
1k
Q2
TIP125
Q1
TIP120
79L05
VI
GND
VO
C4
0.3µF
C5
0.1µF