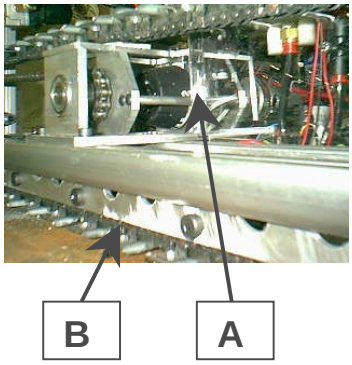
Auto-Balance Gyro Code

The TechnoKats Robotics Team, a partnership between Delphi Automotive Systems and Kokomo High School in Kokomo, Indiana, is proud to be able to share their "Gyro code" from the 2001 FIRST Robotics Competition. This code was a part of a very successful year for the TechnoKats, as their machine, the "TechnoKart" was focused on "auto-balancing" the bridge with two goals. Throughout the year, TechnoKart consistently balanced the bridge during 2 Regionals and the National Competition. At these competitions, TechnoKart auto-balanced two goals with these results:

33 attempts to auto-balance
- 27 successful balances on 1st try
- 2 successful balances on 2nd try
- 4 unsuccessful balances due to a dislodged bridge

#1 seed and 8th place at Nationals, 4th place at W. Mich and Midwest Regionals

The "gyro code" for the Yaw Rate Sensor is shown on page two of this document. This code was a combined with two other features of the robot that made the robot balance when all of them worked together. Power and traction combined with the code to perform the auto-balance, and each feature depended upon each other to do so.



Power: TechnoKart used 4 motors in its drive system. It used tank-style driving using the combination (A) of one Drill Motor and one Fisher-Price Motor to drive each side of the robot. These motors were run through the Drill Motor Gearbox in low gear (while balancing). There was a 3:1 ratio from the drive motors to the 7.5" diameter wheels. This power was able to move a 126 pounds robot with two 80 pounds goals very quickly.

Traction: Special tank treads were created by the TechnoKats in order to get excellent traction. Small steel flanges were welded to 35 chain from SPI. 1/16" thick steel cleats (B) were attached to each set of steel flanges in order to create this chain-based tread which provided superior traction without damaging the carpet.

Gyro: The Yaw Rate Sensor was mounted onto a vibration dampening sheet (SPI: U-NVC-187D) in order to reduce "noise". This sensor detected an angular change and triggered a control loop in the robot's program. The signal from the Gyro was digitally filtered out by a low pass filter to reduce further "noise". The digital low pass filter was implemented as shown in the White Paper of Innovation First entitled "Digital Filter for Joysticks" by Mike Gray.

How the program worked:

Once TechnoKart had the goals on the bottom part of the bridge, our student operator flipped the "auto-balancing switch". He had a choice to flip the switch forward or backward, depending on the direction that the robot was travelling. Once this switch was actuated, the robot went into an autonomous mode and the student drivers had no input to the control of the machine.

The autonomous mode ran the robot through this sequence of operation:

1. Go forward (or backward, depending on switch direction) slowly
2. Yaw rate sensor detects an angular change, sensing the tipping of the bridge.
3. All motors go in the opposite direction for a known count
4. All motors reverse direction for a known count

The amount of distance traveled in the step 3 had to be “tweaked” as we competed on various bridges. Also, the forward-travelling values and backward-travelling values were sometimes different as each bridge balanced differently from one side to the other. Step 4 was needed in order to stop the momentum of the robot and goals, and to get the robot into the center of the bridge.

Code:

```
'---------- FILTER ------------------------------------------------------

sensor1Filt = sensor1Filt + ((256 + sensor1 - sensor1Filt)*KFilt/200) - (256*KFilt/200)
sensor1 = sensor1Filt

'---------- AUTO LEVEL --------------------------------------------------------

IF LevelMode = 1 THEN Check_latch

IF Level_fw = 0 THEN Next_level
       ' Numbers go down when tipped to front
       ' It goes forward until starts tipping
       IF sensor1 < 124 then Stop_level_fw
              Serout USERCPU, OUTBAUD, [255, 255, 155, relayA, 127, relayB, 155, 127, 127,
127, 127, 127, servo_gear, servo_gear+GearOff, 127, 127, BSRollPWM, 127, BSArmPWM, 127]
              Goto MainLoop
       Stop_level_fw:
       LevelMode = 1
       'Back up a little bit
       for k = 1 to 75
              Serout USERCPU, OUTBAUD, [255, 255, 0, relayA, 127, relayB, 0, 127, 127, 127,
127, 127, servo_gear, servo_gear+GearOff, 127, 127, BSRollPWM, 127, BSArmPWM, 127]
       next
       'Then go forward a little bit
       for k = 1 to 50
              Serout USERCPU, OUTBAUD, [255, 255, 160, relayA, 127, relayB, 160, 127, 127,
127, 127, 127, servo_gear, servo_gear+GearOff, 127, 127, BSRollPWM, 127, BSArmPWM, 127]
       next
       ' Then stops completely
       Serout USERCPU, OUTBAUD, [255, 255, 127, relayA, 127, relayB, 127, 127, 127, 127,
127, 127, servo_gear, servo_gear+GearOff, 127, 127, BSRollPWM, 127, BSArmPWM, 127]
Next_level:

IF Level_rv = 0 THEN EndLevel
       ' Numbers go up when tipped to back
       ' It goes Reverse until starts tipping
       IF sensor1 > 136 then Stop_level_rv
              Serout USERCPU, OUTBAUD, [255, 255, 105, relayA, 127, relayB, 105, 127, 127,
127, 127, 127, servo_gear, servo_gear+GearOff, 127, 127, BSRollPWM, 127, BSArmPWM, 127]
              Goto MainLoop
       Stop_level_rv:
       LevelMode = 1
       'Forward a little bit
       for k = 1 to 85
              Serout USERCPU, OUTBAUD, [255, 255, 254, relayA, 127, relayB, 254, 127, 127,
127, 127, 127, servo_gear, servo_gear+GearOff, 127, 127, BSRollPWM, 127, BSArmPWM, 127]
       next
       'Then go Reverse a little bit
       for k = 1 to 50
              Serout USERCPU, OUTBAUD, [255, 255, 95, relayA, 127, relayB, 95, 127, 127,
127, 127, 127, servo_gear, servo_gear+GearOff, 127, 127, BSRollPWM, 127, BSArmPWM, 127]
       next
       ' Then stops completely
       Serout USERCPU, OUTBAUD, [255, 255, 127, relayA, 127, relayB, 127, 127, 127, 127,
127, 127, servo_gear, servo_gear+GearOff, 127, 127, BSRollPWM, 127, BSArmPWM, 127]

Check_latch:
IF (Level_fw = 1) OR (Level_rv = 1) THEN EndLevel
       LevelMode = 0
EndLevel:
```