Engineering Notes

Ahad Rizvi P:5

1. The Raspberry Pi will serve as the central processing unit for the entire system. It will record the accelerometer and gyroscope’s inertial data and run analysis to determine accurate location data. It will use the RFID tag’s data to correct for error.
2. This system here is meant to read and detect the 13.5MHz signal from the RFID reader and send that information to the Pi. When the system is in range of the reader the resonant frequency will be reached and current will flow through to GPIO6 on the Pi. This is one way to approach the RFID sensor fusion. The advantage of this method is that every car will not be required to carry an RFID reader instead will have this resonant circuit to detect the reader’s presence. Another method is described in engineering note 7.
3. This is the resonant circuit. It will act as a voltage divider and only allow the signal to pass through to GPI06 if the signal is the right frequency. It is essentially a bandpass filter. The values have not yet been determined, as I need access to the parts cabinet to run the calculations to accurately tune this circuit to 13.5Mhz.
4. This RFID sends out a 13.5 MHz radio signal. There would only be a few of these per intersection. Each car will only have the resonant circuit described in note 2. It may at some time be beneficial to simply replace this reader with a circuit that generates a 13.5Mhz radio signal, for cost and simplicity.
5. This is the accelerometer currently in use in the circuit. It is a very imprecise tool, and requires much analysis to even interpret the data in any successful manner. Code detailing the analysis has been included. This may at sometime be replaced with a better, more accurate, yet more expensive serial accelerometer.
6. The accelerometer plus the gyroscope provide five dimensions for movement. The x,y,z coordinates provide location in space, and theta and phi provide rotational information. In reality a vehicles motion has 6 degrees of freedom. Yet the third rotational coordinate, which would only drastically change if the vehicle began to tumble, is not really necessary.
7. This alternate approach to RFID data involves attaching a reader to the PI. The reader will read RFID tags placed throughout the intersection and transmit information back to the PI when the data was read. This situation is more expenseive, and a tad excessive, but it has been included, because later testing may determine that the system described in engineering note 2 may be more difficult to work with.
8. This offset value is the default rest value for the x-dimension accelerometer value. Deviance from this value determines acceleration.
9. The data is currently being read from an Arduino, as the code is running on my computer. Once integrated into the PI, the pi will read the accelerometer straight.
10. If the acceleration value is minimal, this is likely extraneous noise. Here the noise is filtered out.
11. This is the left-hand-rule approximation process here. The acceleration data is integrated, using the left-hand-rule. The velocity data is again integrated to provide the position data.
12. If the system is accelerated for a brief period, the velocity curve should rise then fall then hit zero. Testing showed that, due to impreciseness in the accelerometer this almost rarely happened perfectly. Instead the velocity would return to some resting value that was almost zero, this would have exponentiated effects on the position data. This section of code checks if the velocity data has remained constant for 100 continuous measurements and recalibrates the velocity value to zero.
13. The data is written to a log file for analysis.
14. This code process data for one dimension. This code can be expanded easily to multiple dimensions, once the preliminary error correction and analysis for one dimension is suitable enough.