Detektion von IR-Signalen

Ingo Boersch, 12/2007

Problembeschreibung

Bei dem Roboterwettbewerb Hasenjagd sendet der Hase ein mit 100 Hz moduliertes Infrarotsignal, während der Fuchs mit 125 Hz moduliert. Dabei ist dem Empfänger der eigene Sender sehr nah, der eigentlich interessante Sender des Gegners jedoch weiter weg. Maßnahmen den Empfänger vor der Strahlung des eigenen Senders abzuschirmen führten nicht zum Erfolg, da viele Reflexionen vorhanden sind. Beide Signale überlagern sich und werden vom Fernsteuerempfänger an einem digitalen Eingang des AKSEN-Boards aufgenommen. Die Infraroterkennungsroutine der AKSEN-Bibliothek hat Schwierigkeiten, in dem Signalgemisch die einzelnen Frequenzen zu erkennen, so dass die entsprechenden Zähler oft auf 0 zurückfallen. Die Detektion eines einzelnen Signals funktioniert ohne Probleme bis ca. 3m.

Lösungsidee

Die Nachbildung eines Signalgemischs einer 100Hz- und 125Hz-Schwingung mit variablem Offset und Abtastrate 1ms in Excel zeigte bei der Auswertung mit der Excel-internen Fouriertransformation für alle Offsets im Frequenzspektrum deutliche Spitzen bei den beteiligten Perioden 8ms und 10ms. Es liegt also nahe, eine vereinfachte FT zur Erkennung zu implementieren.

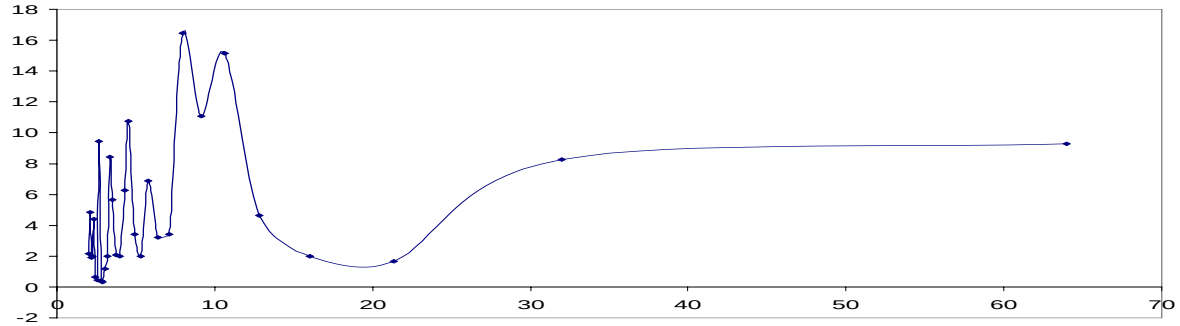

Abbildung 1: Amplitudenspektrum eines Signalgemisches mit Offset(10ms)=2 und Offset(8ms)=0, erzeugt mit der Excel-internen FT: Deutlich sind die beiden Maxima um 8 und 10ms zu erkennen

Für den Realteil der Fouriertransformierten eines reellen Signals für die Frequenz k gilt:

$$\mathrm{Re}(X(k)) = \sum_{n=0}^{N-1} x(n)\cos(-2\pi kn/N)$$, dito für den Imaginärteil:

$$\mathrm{Im}(X(k)) = \sum_{n=0}^{N-1} x(n)\sin(-2\pi kn/N)$$

mit x(n) = Signal an der Stelle n
N = Länge des betrachteten Signals, wir wählen N=40 ms=40 Takte
k = 5 bzw. 4 Schwingungen pro N Takten, also 100 bzw 125 Hz

Die Amplitude einer Frequenz berechnet sich als $A(k) = \sqrt{\mathrm{Re}(X(k))^2 + \mathrm{Im}(X(k))^2}$

Die vom Signal unabhängigen Ausdrücke können vorher berechnet werden, wir bezeichnen sie als Steuervektoren:

$re8(n) = \cos(-\pi n/4)$

$im8(n) = \sin(-\pi n/4)$

$re10(n) = \cos(-\pi n/5)$

$im10(n) = \sin(-\pi n/5)$

Die Steuervektoren sind sinusförmige Schwingungen, die zueinander um 90° versetzt sind.

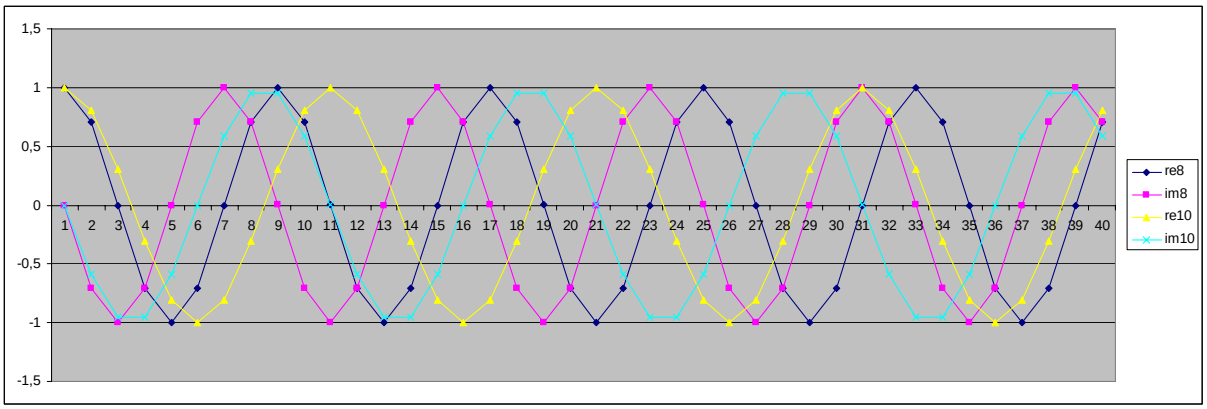

Abbildung 2: Steuervektoren einer FT: Zu erkennen sind Sinus und Cosinus jeweils mit 4 oder 5 Perioden innerhalb von 40 Takten

Bei der Suche wird später zusätzlich auch die negative Korrelation berücksichtigt, so dass die Phasenlage in 90°-Schritten abgesucht wird. (sin, cos, -sin, -cos)

Die elementweise Multiplikation der Steuervektoren mit dem Signal und die folgende Addition und Betragbildung führt zu deutlichen Amplitudenwerten A(4) und A(5) bei N=40 und beliebigen Offsets der beiden Signale (Variante 2: Eigene FFT für 2 Frequenzen N=40).

Die Verkürzung der untersuchten Signallänge auf 20 Takte (Variante 3: Eigene FFT für 2 Frequenzen N=20) erzeugte bei einigen Offsetkonstellationen hohe Amplituden, obwohl kein Signal vorhanden war und niedrige Amplituden bei vorhandenem Signal. Hier ist der 2D-Raum möglicher Offsets zu untersuchen, um ggf. eine noch brauchbare minimale Signallänge zu bestimmen. Wir bleiben erstmal bei N=40.

Umsetzung auf dem AKSEN-Board

Zur Umsetzung auf dem AKSEN-Board finden folgende Vereinfachungen Verwendung:

1. Ganzzahl-Arithmetik: Die reellen Elemente der Steuervektoren stammen aus dem Intervall [-1,1] und werden durch Multiplikation mit 100 und Runden auf ganze Zahlen in [-100,100] abgebildet (Variante 4: Eigene FFT mit Ganzzahlen für 2 Frequenzen N=40).
2. Die Summe der Quadrate wird ersetzt durch die Summe der absoluten Beträge. Dies genügt, wenn aus den Daten nur die binäre Aussage „Empfang oder nicht“ abgeleitet wird.

Es ergibt sich der angehängte Quelltext, der noch viel Optimierungspotential enthält.

Mögliche Optimierungen

1. Da die Signalwerte nur 0 oder 1 annehmen, kann die Multiplikation durch eine bedingte Addition ersetzt werden.
2. Fehler?: In der Excel-Tabelle wird von einem Signal im Bereich [-1, 1] ausgegangen, die Implementierung verwendet [0,1]. Bei signal(n)==0 ist daher eigentlich das Element des Steuervektors zu subtrahieren.
3. Statt der sinusförmigen Steuervektoren zeigt eine Simulation mit Rechtecksignalen (Variante 5: Rechteckige Suchfolgen für 2 Frequenzen N=40) eine ebenso gute Erkennungsleistung, so dass statt Addition (Optimierung 1) nur noch Inkrement und Dekrement anfallen
4. Steuervektoren benötigen keine Integerelemente
5. Zugriff auf Arrayelemente (z.B. signal(n), Steuervektoren) durch einen laufend inkrementierten Pointer ersetzen
6. Die Berechnung möglichst schon in der Aufzeichnungszeit vornehmen. Dabei muss aber garantiert werden, dass jede Millisekunde abgetastet wird.
7. Aufzeichnungszeit N verringern, bis die Simulation Fehler erster oder zweiter Art zeigt.

Anlage: Excel-Datei mit Varianten zur Simulation der Erkennung

Praktische Erfahrungen und Verbesserungen senden an boersch@fh-brandenburg.de

```
int   int_re1_steuer[]=   {100,71,0,-71,-100,-71,0,71,100,71,0,-71,-100,-71,0,71,100,71,0,-71,-
100,-71,0,71,100,71,0,-71,-100,-71,0,71,100,71,0,-71,-100,-71,0,71};
int    int_im1_steuer[]=    {0,-71,-100,-71,0,71,100,71,0,-71,-100,-71,0,71,100,71,0,-71,-100,-
71,0,71,100,71,0,-71,-100,-71,0,71,100,71,0,-71,-100,-71,0,71,100,71};
int    int_re2_steuer[]=    {100,81,31,-31,-81,-100,-81,-31,31,81,100,81,31,-31,-81,-100,-81,-
31,31,81,100,81,31,-31,-81,-100,-81,-31,31,81,100,81,31,-31,-81,-100,-81,-31,31,81};
int   int_im2_steuer[]={0,-59,-95,-95,-59,0,59,95,95,59,0,-59,-95,-95,-59,0,59,95,95,59,0,-59,-
95,-95,-59,0,59,95,95,59,0,-59,-95,-95,-59,0,59,95,95,59};

int signal[40];
#define IR_PROC_WAIT 40
#define IR_OK 150

void ir_proc(void)
{
       int re1, im1, re2, im2;
       int empf_8ms,empf_10ms;
       unsigned char n;

       unsigned long alte_ms, start;
       int ms;

       re1 = im1 = re2 = im2 = 0;

       motor_richtung(3,1);
       motor_pwm(3,10);

       // exclusive busy waiting
       motor_richtung(3,1);
       process_hog();
       alte_ms = akt_time();
       start = akt_time();
       ms = 0;
       // 40 ms Signal aufzeichen
       while (ms < IR_PROC_WAIT)
       {
              // auf beginn einer neuen ms warten
              while (akt_time() == alte_ms);
              signal[ms]=digital_in(4);
              ms++;
              alte_ms = akt_time();
       }

       motor_pwm(3,0);

       //FFT berechnen
       // re1,im1 für 8ms und re2, im2 fuer 10ms
       for(n=0;n<40;n++) {
              re1 += signal[n]*int_re1_steuer[n];
              im1 += signal[n]*int_im1_steuer[n];
              re2 += signal[n]*int_re2_steuer[n];
              im2 += signal[n]*int_im2_steuer[n];
       }

       // Summe der absoluten Beträge, richtig wäre Summe der quadrate, aber die zahlen wären
zu gross
       if (re1 <0) re1=-re1;
       if (im1 <0) im1=-im1;
       if (re2 <0) re2=-re2;
       if (im2 <0) im2=-im2;
       empf_8ms = re1 + im1;
       empf_10ms = re2 + im2;

       lcd_cls();
       lcd_uint(empf_8ms);
       lcd_setxy(1,1);
       lcd_uint(empf_10ms);

       // Anzeige mit LED
       if (empf_8ms >= IR_OK )  led(3,1); else led(3,0);
       if (empf_10ms>= IR_OK )  led(2,1); else led(2,0);

       sleep(200);
}
```