Lookup and Interpolate

Nichtlineare Funktion von Sensoren

Viele Sensoren, aber auch andere Funktionen, arbeiten extrem nicht-linear. Beispiel:
PT100, PTC, NTC, Foto-Detektoren, aber auch z.B. Dioden. Die Reihe lässt sich fast beliebig fortsetzen. Da dies alles analoge Werte sind, werden diese in der Regel mit einem AD-Wandler erfasst. Normalerweise steht das Messergebnis in einem bestimmten Verhältnis zu dem äusseren Ereignis, z.B. Temperatur. Aber eben dieses Verhältnis ist sehr oft nicht linear. Ein PT100 z.B. hat bei 0grad einen Widerstand von 100 Ohm bei 50grad ca. 124 Ohm, bei 100grad ca. 143 Ohm. Der Zusammenhang zwischen Temperatur und Widerstand ist nicht-linear.

Um jetzt die aktuelle Temperatur zu erhalten, kann man zwei Wege beschreiten:

1. Mit einer passenden Formel, die meistens sehr komplex ist, kann die dem Widerstandswert zugeordnete Temperatur ermittelt werden.
2. Man erstellt sich eine sogenannte LookUp-Table in der in Stufen bestimmte Widerstandswerte und die zugehörige Temperatur abgelegt sind. Mit dem Widerstanswert (Messwert) als Index wird jetzt auf die Tablle zugegriffen. Kann das Messergebnis von 100 bis 200 laufen, werden 100 Werte (Ergebnisse)in der Tabelle benötigt. Etwas schwieriger wird es, wenn der Analogteil und AD-Wandler so beschaltet ist, dass das Messergebnis von 0 bis 1023 variiert. Dann braucht man 1024 Einträge in der Tabelle.

Linearisierung

Die vorliegende Implementation basiert auf einer Tabelle, in der sowohl die Mess-Ergebnisse als auch die zugehörigen Resultate immer paarweise abgelegt sind. Der LookUp Algorithmus sucht mit dem Messergebnis als Argument in der Tabelle, bis er auf einen gleichen Wert stösst, oder mit dem Argument zwischen 2 Werten liegt. Der Suchvorgang wird aus Geschwindigkeitsgründen mit einem binären Verfahren ausgeführt.

Ist ein passender Wert gefunden, wird das Ergebnis zurückgegeben. Liegt das Argument zwischen 2 Werten, wird linear interpoliert. Dieses Verfahren ermöglicht je nach Genauigkeits Forderungen relativ kurze Tabellen. Wenn die Koordinaten Zahl in der Tabelle relativ hoch ist, ergibt die lineare Interpolation eine ausreichende Genauigkeit, da man im allg. davon ausgehen kann, dass kurze Abschnitte auch näherungsweise linear sind.

System Funktionen:

```
function InterPolX(const LookUp : pointer; x : integer; var y : integer) : boolean;
function InterPolX(const LookUp : pointer; x : longint; var y : longint) : boolean;
function InterPolX(const LookUp : pointer; x : float; var y : float) : boolean;

function InterPolY(const LookUp : pointer; y : integer; var x : integer) : boolean;
function InterPolY(const LookUp : pointer; y : longint; var x : longint) : boolean;
function InterPolY(const LookUp : pointer; y : float; var x : float) : boolean;
```

Der Pointer muss in die Tabelle im ROM zeigen. Das erste Argument ist der Suchwert. Er bestimmt auch die Arbeitsweise (Integer, LongInt oder Float) der Funktion. Das Ergebnis wird im zweiten Parameter abgelegt, wenn die Suche erfolgreich war.

LookUp Table Definition und Import

Der Parameter **const LookUp : pointer** der den Funktionen übergeben werden muss, muss ein Pointer sein, der auf die im ROM liegende Tabelle zeigt. Diese Tabelle muss so konstruiert sein:

```
const
  IntLookUp   : array[1..(size * 4) + 3] of byte = FileName;
  // size point.x point.y of integer, 3bytes info

  LongLookUp  : array[1..(size * 8) + 3] of byte = FileName;
  // size point.x point.y of longint, 3bytes info

  FloatLookUp : array[1..(size * 8) + 3] of byte = FileName;
  // size point.x point.y of float, 3bytes info
```

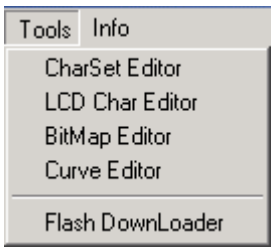
Die Array Definition ist eigentlich nur proforma als Platzhalter für das zu ladende binär File gedacht. Der Parameter **Size** bestimmt die Anzahl der x/y-Datenpaare und der dahinter stehende Multiplikationsfaktor (4 bzw. 8) ist die Anzahl der Bytes für einen Datenpaar (sizeOf(integer), sizeOf(LongInt) oder sizeOf(float)). Die drei Bytes Info enthalten den Datentyp und die Datenpaar Zahl.

Erzeugung der LookUp Table

Zur Erstellung der Tabelle enthält das System den Table Generator **CurveGen**, mit dem man graphisch und interaktiv eine Kurve erstellen und als binär File zum Import in die Applikation ablegen kann.

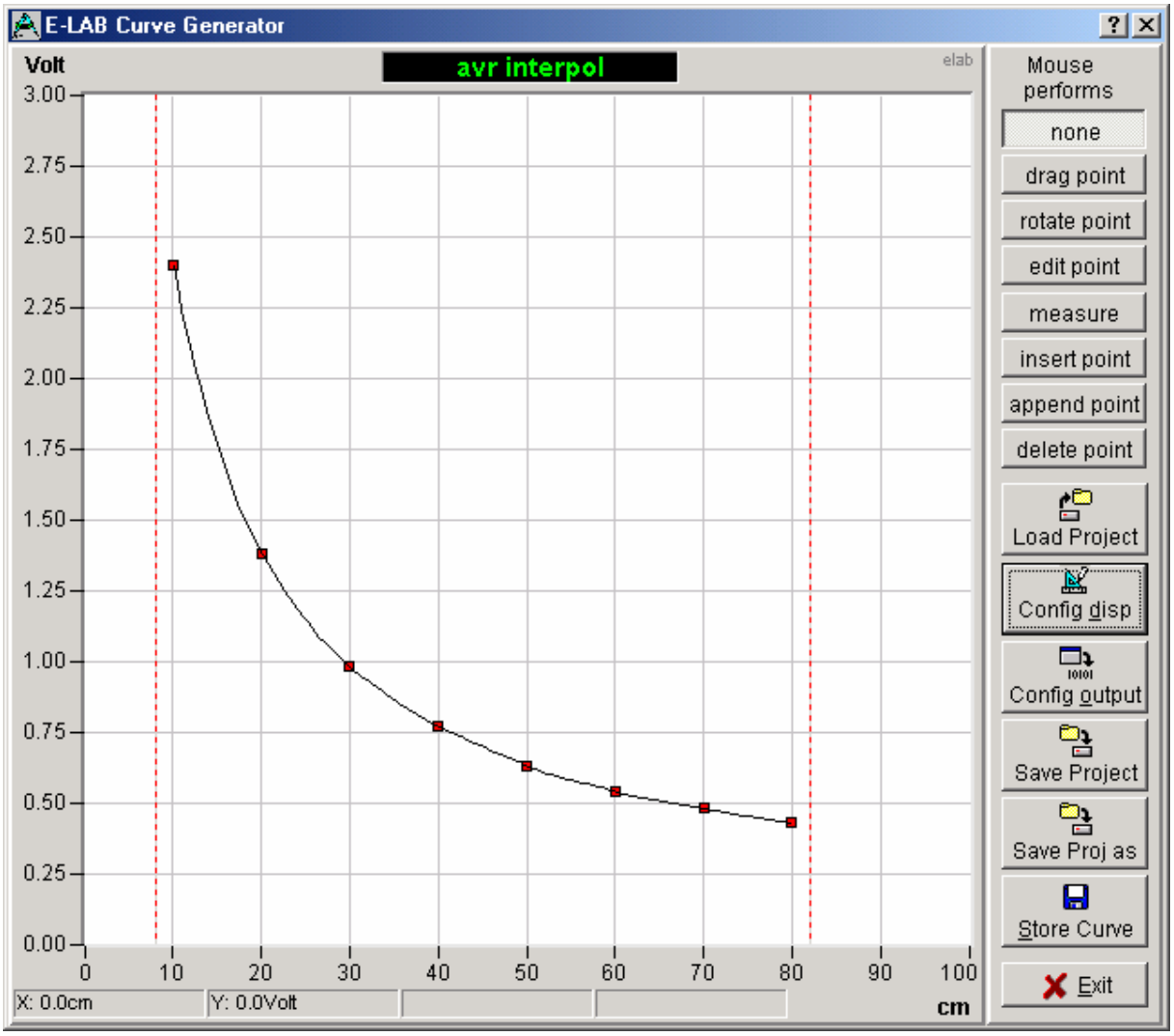
Ein Beispiel Programm ist unter den Demos als "AVR Interpol" zu finden. Hierin wird ein optischer Entfernungsmesser (Sharp) linearisiert und die Entfernung in cm ausgegeben. Ein Datenblatt dieses Sensor ist ebenfalls in der Applikations Directory zu finden.

Programm Aufruf



Das Programm *CurveGen.exe* wird aus der IDE PED32 heraus mit dem Button  oder mit nebenstehendem Menu aufgerufen. Es generiert nach Erfassung der Parameter und Stützpunkte das für die Interpolation benötigte binär File.

CurveGen ist vollkommen interaktiv und graphisch ausgelegt. Dadurch erhält man immer eine optische Aussage für jede Manipulation der Parameter. Die Teilkurven zwischen den vorgegebenen Stützpunkten werden nach einem speziellen, Spline ähnlichen Verfahren errechnet. Damit ist es auch möglich treppenförmige Kurven zu erzeugen, was mit einem reinen Spline nicht möglich ist.



Das obige Bild zeigt einen Screenshot von *CurveGen* mit dem geladenen Projekt **AVR Interpol**.
Die beiden senkrechten roten Linien bezeichnen dabei die Anfangs und Endpunkte für die Table Generierung.
Die roten Quadrate sind die vorgegebenen Stützpunkte.



Die obere Hälfte des nebenstehenden Bildes zeigt die Einstellmöglichkeiten für Maus Clicks und Bewegungen.
None bedeutet, wie der Name sagt, dass die Maus keine spezielle Funktion im Display hat.
Drag point verschiebt einen angeclickten roten Stützpunkt, solange die linke Maustaste gedrückt bleibt.
Rotate point erlaubt durch anclicken und ziehen mit der Maus einem Punkt eine bestimmte Richtung (Vektor) und Gewichtung zu geben.
Mit **edit point** wird durch einen Click auf einen Stützpunkt ein Dialog eröffnet, mit dem man die exakte Position (x/y) des Punktes eingeben kann.
Measure erlaubt durch rechten bzw. linken Mausclick zwei Fadenkreuze zu verschieben, die als Messpunkte dienen. Die Resultate werden am unteren Rand eingeblendet.
Mit **Insert point** können durch Mausclicks zusätzliche Stützpunkte zwischen zwei vorhandenen eingesetzt werden.
Append point hängt mit jedem Mausclick einen Stützpunkt an den letzten rechten an.
Mit **delete point** kann ein vorhandener Punkt mit einem Mausclick gelöscht werden.

Project admin öffnet den weiter unten beschrieben Projekt Dialog.

Config display öffnet den Dialog zum einstellen der Display Parameter wie Skalierung, Teilung etc. Beschreibung siehe weiter unten.

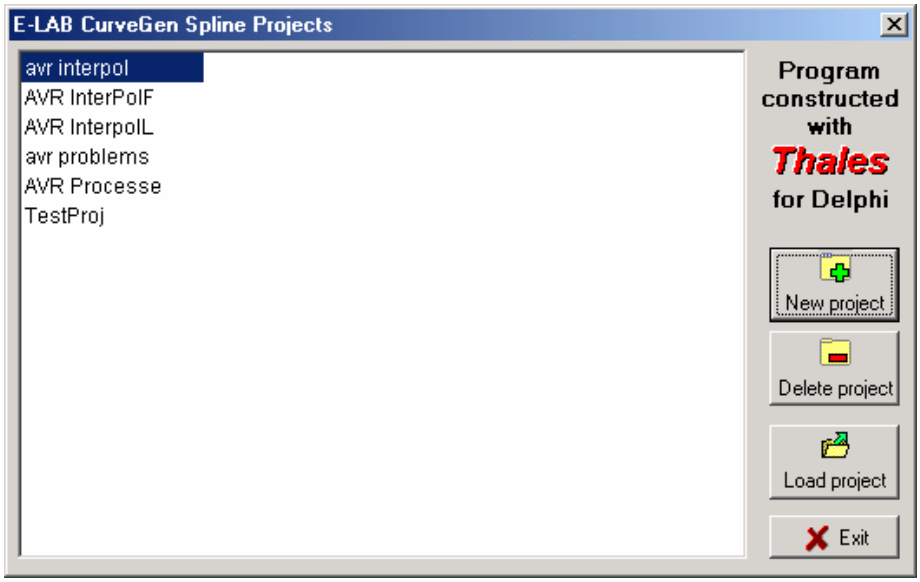
Config output öffnet den Dialog zum Editieren der Ausgabe Parameter, wie z.B. Datenformat (integer, longint etc). Beschreibung siehe weiter unten.

Mit **Save Project** werden die aktuellen Projekt Daten und Einstellungen in der Projekt Datei *ProjectName*.inicg abgelegt.

Mit **Save Proj as** kann das aktuelle Projekt unter einem anderen Namen abgelegt werden.

Store Curve öffnet den weiter unten stehenden Export Dialog. In einem Fenster sind dabei im Klartext die generierten Tabellen Daten zu sehen, mit einem Überblick zu den Einstellungs Vorgaben.

Exit beendet das Programm.

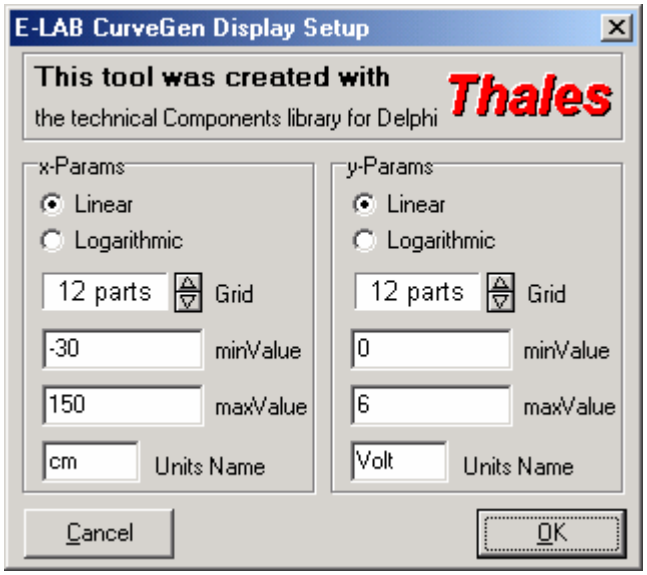


Der Projekt Auswahl Dialog bietet neben dem Laden schon vorhandener Projekte auch Projekte zu löschen als auch neue Projekte anzulegen.

Mit einem Doppelclick auf einen Eintrag wird dieses Projekt geladen.

Mit einem Click auf den **New project** button erscheint ein Dialog zur Eingabe des neuen Projekt Namen und und des Arbeits Verzeichnisses.

Ein neues Projekt enthält schon eine Default Einstellung und eine kleine Zahl von passenden Stützpunkte. Sämtliche Einstellungen müssen jetzt den Vorgaben (Sensor etc) entsprechend angepasst werden.



Mit dem **Display** Setup Dialog wird die Skalierung und Teilung des Displays eingestellt.

Die Einstellung erfolgt für die x und y Achse getrennt. Es kann zwischen linearer und logarithmischer Darstellung gewählt werden.

Die Einstellung der Skalenteilung muss so gewählt werden, dass bei der Skalenbeschriftung keine Rundungsfehler auftreten.

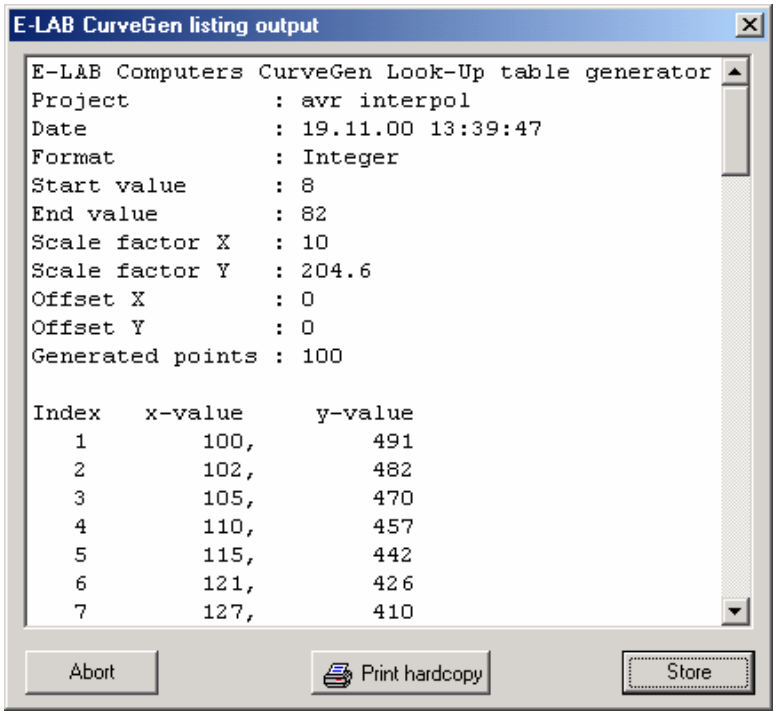
Die Minimal Werte der Skalierung beziehen sich auf die untere linke Ecke, die Maximal Werte auf die obere rechte Ecke des Displays.



Mit nebenstehendem **Export** Dialog werden die Vorgaben für die Erzeugung des binär Files erfasst.
Der Werte Typ wird unter **Exported type** eingestellt.
Mit **Ranges** wird ein Ausschnitt aus der Kurve ausgewählt, ‚der im Display mit den roten senkrechten Linien dargestellt wird. Start und End Value beziehen sich dabei auf die x-Achse. Es werden dabei nur die Punkte exportiert, die innerhalb dieser X-Grenzen liegen.
Der Wert **point count** bestimmt wieviele Werte innerhalb des gewälten Bereiches im binär File abgelegt werden sollen. Muss geradzahlig sein.

Für die x und y Werte können **Skalierungen** und Offsets für den Export angegeben werden. Die Berechnung sieht dabei so aus:

```
val:= (val – offset) * factor
```



Der **Output** Dialog listet alle relevanten Vorgaben und die generierten Wertepaare auf . Diese Liste kann auch ausgedruckt werden.

Mit dem **Store** Button wird das Binär File *ProjectName*.crvg erzeugt, das direkt in die Source nach oben beschriebenen Schema importiert werden kann.

Eigenschaften von CurveGen

Genauigkeit:

Das Programm basiert auf einer speziellen Spline Funktion. Spline hat die Eigenschaft, dass die generierte Kurve immer durch ihre Stützpunkte geführt wird. Das bedeutet in der Konsequenz, dass jede Ungenauigkeit bei der Erfassung der **Stützpunkte** zumindest bei diesem Punkt voll und in seiner näheren Umgebung zum Teil durchschlägt. Damit wird die Präzision der interpolierten Werte zum ganz überwiegenden Teil von den Stützpunkten abhängig.

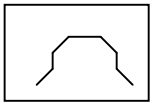
Nicht ganz so gravierend für die Genauigkeit ist die **Anzahl** der zur Verfügung gestellten Stützpunkte. Bei „normalen" quadratischen, kubischen oder logarithmischen Kurven kommt man mit relativ wenigen Punkten aus (5..20). Je komplexer die Kurvenform ist, desto mehr Stützpunkte braucht man. Der Extremfall wäre eine logarithmische Kurve mit linearen Teilen und Treppen.

Kurvenform:

Durch die Zahl und Lage der Stützpunkte ist die Kurvenform im Normalfall schon sehr gut vorgegeben. Eine unbearbeitete neue Kurve hat noch eine strenge Konzentration der Richtungen und Gewichte auf den einzelnen Stützpunkten. Das ist leicht daran zu erkennen, dass die Teilstücke zwischen den Punkten linear verlaufen.

Dieser Effekt ist aber normalerweise unerwünscht. Eine Kurve sollte auch eine Kurvenform haben. Deshalb ist es unumgänglich die Attribute der Stützpunkte richtig einzustellen. Dies geschieht mit der Maus und der Einstellung **rotate point**. Mit gedrückter linker Maustaste wird der Punkt gezogen.

Es bewegt sich jetzt allerdings nicht der Punkt, sondern visuell seine beiden Attribute in Form eines weissen Kreises. Durch verschieben des weissen Kreises wird die Gewichtung und die Richtung (Vektor) des Punktes verändert. Je grösser die Distanz des Kreises vom Punkt ist, desto mehr wird die Kurve in diese Richtung verzogen. Diese Einstellung muss ebenfalls sehr sorgfältig vorgenommen werden, damit die erzeugte Kurve möglichst nahe an die Ideal Kurve herankommt. Das menschliche Auge ist in dieser Richtung relativ unbestechlich.



Eindeutigkeit: Kurven müssen aus der Sicht des LookUp Parameters eindeutig sein. Im nebenstehenden Bild würden sich bei einer Vorgabe des y-Wertes zwei x-Werte ergeben. Welcher jetzt von der Funktion zuruckgegeben wird ist unbestimmt. Ähnliches gilt auch umgekehrt für den x-Vorgabe Wert. Sätere Implementation der AVRco LookUp Funktion werden eine Bereichsvorgabe für die LookUp Funktionen erlauben.

Generierung:

Ist die Kurve fertig konfiguriert, so kann das binär File erzeugt werden. Hierzu sollten noch die Skalierungs Faktoren, Offset, Anzahl der Punkte und der Kurven Ausschnitt vorgegeben werden. (**Export** Dialog)

Der **Ausschnitt** aus der Kurve kann fast beliebig gewählt werden. Es ist allerdings nicht sinnvoll, die Grenzen ausserhalb der x-Anfangs und Endwerte zu legen. In diesem Fall werden nur Punkte innerhalb der x-Werte der Kurve exportiert. Ein zu kleiner Ausschnitt bringt später bei der Interpolation u.U. relativ ungenaue Werte.

Die Offsets und Skalierungs Faktoren müssen so gewählt werden, dass im Programm brauchbare Werte herauskommen. Wird der y-Wert zum Beispiel durch einen10bit AD-Wandler gebildet, sollte sich das Resultat aus ((Kurven-y-Wert – offset) * factor) auch im Bereich von 0..1023 bewegen. Ähnliches gilt auch für den x-Wert. Die Werte von **offset** unf **factor** sind erheblich abhängig von der Gesamt Verstärkung des System, das heist von der Kette vom Sensor über OP-Amps bis zum AD-Wandler und dessen Auflösung.

Die Anzahl der generierten Punkte wird durch die Vorgabe im Dialog Feld **point count** bestimmt. Hier muss ein Kompromiss eingegangen werden. Wünschenswert wäre eine möglichst grosse Zahl von Punkten, so dass der durch das lineare Interpolieren des Systems erzeugte Fehler möglichst Null wird. Dies bedeutet allerdings sehr grosse Tabellen mit entsprechend langer Suchzeit. Wieviel Punkte wirklich gebraucht werden, kann nur ein Praxis Test erweisen. Eine Obergrenze ist die Auflösung des Erfassungs Systems, z.B. AD-Wandler. Mit 10bits sind nun einmal nicht mehr als 1024 mögliche Eingangswerte möglich.

Grundsätzlich ist bei Genauigkeits Überlegungen zu fragen: welches Teil im System ist das ungenaueste und lässt sich auch nicht verbessern. Meistens fällt da der Sensor ins Auge. Wenn dieser grosse Toleranzen hat, macht es wenig Sinn innerhalb der Software auf 0.1% Genauigkeit hinzuarbeiten.