Autor: Kurt Diedrich

Software

Wichtige Hinweise zu den hier vorgestellten Programmen

Die hier gezeigten Listings wurden in Visual Basic 2005 geschrieben, dessen Vorgängerversion auch als VB.net („VB dot net“) bekannt ist. Quellcodes und Exe-Dateien, die in VB 2005 geschrieben wurden, sind nur auf Windows-Rechnern editierbar und lauffähig, auf denen „.net framework“ („dot net framework“) installiert ist. Dies dürfte bei allen Rechnern mit Software neueren Datums jedoch weitgehend der Fall sein.

Bei den folgenden Programmzeilen handelt es sich **nicht** um die Software, mit der die in diesem Beitrag gezeigten Bilder berechnet wurden, sondern um einfache und „abgespeckte“ Versionen mit reduziertem Code und vielen Kommentaren. Diese Versionen sind leicht verständlich, zum Experimentieren geeignet und lassen sich flexibel und nach eigenen Ideen erweitern.

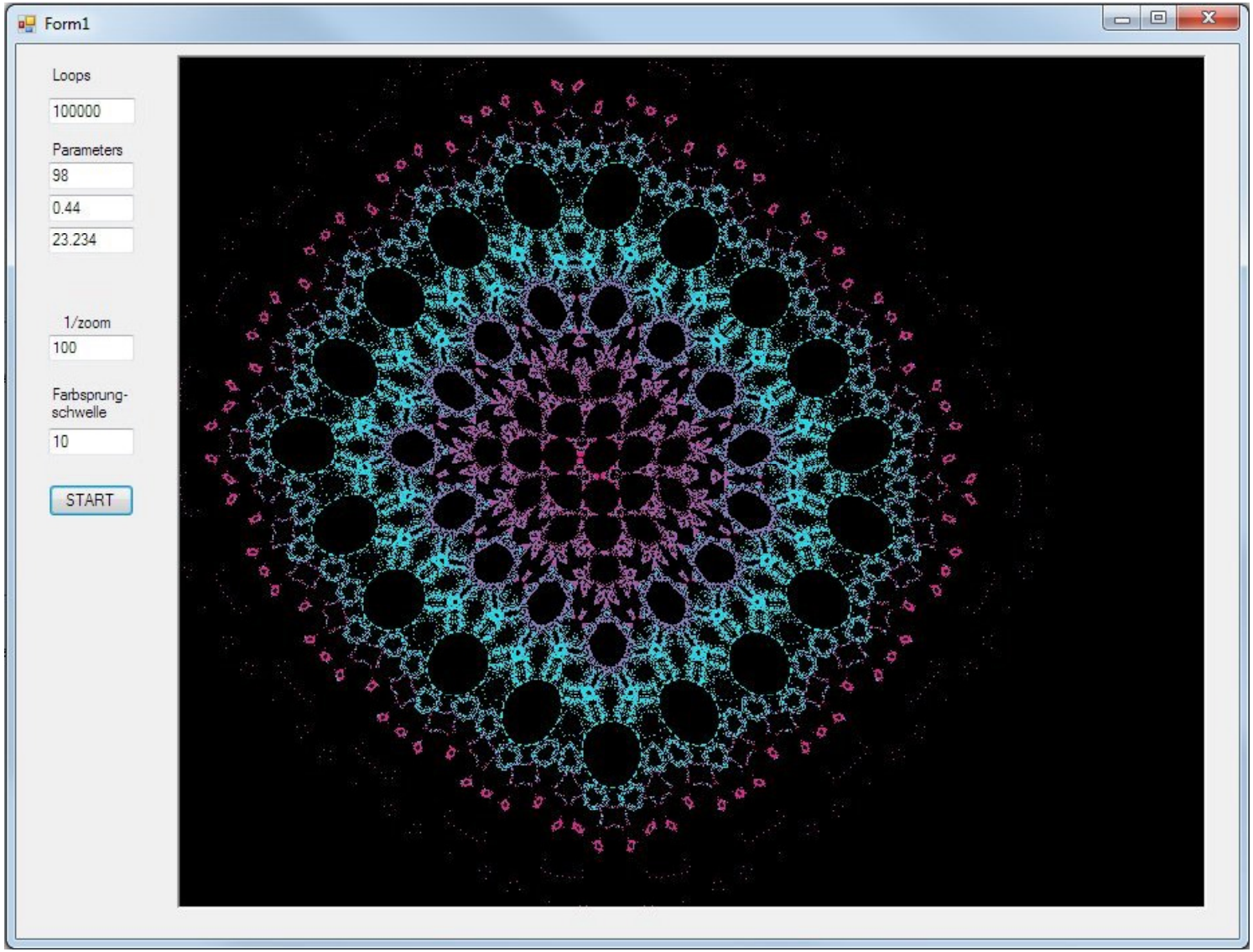
Wer sich mit Visual Basic 2005 auskennt, wird ohne große Mühe in der Lage sein, den Code zum Beispiel mit Copy und Paste in eine VB-Form zu übertragen und diese anhand der dazugehörigen Screenshots der Benutzeroberfläche (VB-Form) entsprechend einzurichten.

Auch für Nicht-Programmierer werden die Listings von Interesse sein, da sie zeigen, worauf es bei den Berechnungen und den einzelnen Schleifen ankommt.

Im Folgenden zunächst ein etwas verschlanktes, klassisches Hopalong-VB-Programm zum Zeichnen von „Häkeldeckchen“:

Voraussetzung für alle folgenden Programme: Kenntnisse in VB-Programmierung

1) Hopalong Algorithmus konventionell



Das hier vorgestellte Programm zeichnet Figuren nach dem Hopalong-Algorithmus, die an Querschnitte durch Pflanzen oder an Häkeldeckchen erinnern. Es bildet die Grundlage des zweiten, hier vorgestellten Programms, Hopalong-Special. Wer versteht, wie die hier gezeigte, einfache Version funktioniert, wird auch die Funktionsweise der Special-Version verstehen.

Wer das Programm mit VB 2005 (bz. VB 2010) selber „nachbauen“ möchte, muss die Form (Oberfläche) zunächst mit den oben gezeigten Objekten füllen: Fünf TextBoxen, ein Button, eine PictureBox. Um die zu den einzelnen Subroutinen gehörenden Codes per Copy und Paste von dieser Stelle in das Original-VB-Listing zu übertragen, müssen der Button und die Form zuerst angeklickt werden. In das sich daraufhin öffnende Codefenster kann der zum jeweiligen Objekt gehörende Code dann hineinkopiert werden. Das Programm funktioniert nicht, wenn man den Code an einem Stück ins Codefenster kopiert.

Bedienung
Die Parameter sind so voreingestellt, dass das im Bild gezeigte Objekt direkt nach dem Anklicken des Buttons „Start“ gezeichnet wird. Durch Überschreiben der in den TextBoxen „Parameters“ dargestellten Werte lassen sich andere Figuren erzeugen. Nicht alle Werte liefern gleich gute Ergebnisse. Hier ist etwas Geduld und Experimentierfreude gefragt. Es sind auch negative Zahlen und mehrere Nachkommastellen möglich.

Durch Ändern der Werte in der Box „Loops“ lassen sich auch Schleifenzahlen bis in den Millionenbereich realisieren. Erfahrungsgemäß kann das Programm in solchen Fällen mehrere Minuten lang beschäftigt sein – auch ohne dass man etwas sieht: Es werden gelegentlich Punkte in Bereiche außerhalb des Sichtfeldes gezeichnet, so dass der Eindruck erweckt wird, das Programm sei abgestürzt. In diesem Falle sollte man einfach eine Weile warten. Dass das Programm fertig ist, erkennt man daran, dass sich der Mauscursor beim Bewegen über die TextBoxen von einem Pfeil in einen Schreibcursor verwandelt.

Leider besitzt das Programm zurzeit noch keinen vernünftigen Fehlerabfang, so dass ein Klicken mit der Maus neben das Programmfenster oder ein Verschieben des Programmfensters während des Zeichnens leider zu einem Absturz führt.

Der eingestellte Zoom-Wert wird reziprok übernommen und muss größer als 1 sein: Höhere Werte führen zu einer Verkleinerung der Abbildung. Diese Eingabe ist notwendig, da, je nach eingegebenem Parameter, die Größe der berechneten Objekte sehr unterschiedlich sein kann.

Durch Eingabe verschiedener Werte in „Farbsprung-Schwelle“ kann festgelegt werden, wie schnell sich die Farben bei Zeichnen ändern.

Programmbeschreibung:

Das Programm beginnt mit der Variablendeklaration. Anschließend erfolgt ein automatisches „Füllen“ der auf der Oberfläche vorhandenen TextBoxen mit Standardwerten, damit man nach dem Start durch Klicken des Startknopfes gleich loslegen kann. Das Ganze erfolgt in einer Sub mit der Bezeichnung „Form_Load“, die beim Anlegen eines neuen Projektes automatisch vorgegeben wird und sich durch Klicken auf die leere Form öffnet.

Die Sub mit dem Namen „Button1_Click“ wird beim Anklicken des Start-Buttons ausgeführt. Sie enthält alles, was zum Zeichnen der Hopalong-Figur benötigt wird:

Zunächst müssen einige Parameter auf Null gesetzt werden, damit sich ihre Werte bei einem erneuten Start nicht akkumulieren und überlaufen.

Es folgt der zum Löschen des Bildschirms (der PictureBox) erforderliche VB-Befehl.

Anschließend werden den 6 aufgelisteten Variablen die in den TextBoxen stehenden Werte zugewiesen. Dadurch lassen sich vor dem Zeichnen neuer Figuren immer wieder neue Parameter eingeben.

Mit „For i = …“ folgt die eigentliche Iterationsschleife. Die Variable „schleifenzahl“ gibt die Anzahl der Iterationsschritte vor.

Der Befehl zum Zeichnen von Rechtecken, Linien oder Kreisen ist in VB 2005 bzw. VB 2010 leider sehr kryptisch. Erfahrene Programmierer werden erkennen, dass es sich bei den am Ende der Zeile stehenden Werten um die X- und Y-Koordinaten zum Zeichnen handelt. Durch die Schreibweise „X/teilfaktor+300“ wird dafür gesorgt, dass die Objekte mit der richtigen Größe an die richtige Stelle gezeichnet werden, da ansonsten die Mitte des Hopalong-Objektes in der linken, oberen Ecke des Bildschirms liegen würde.
Liegt die Figur zu weit links oben oder rechts unten, so kann die Konstante „300“ entsprechend angepasst werden. Diese Konstante kann natürlich auch als weitere Variable über eine zusätzliche TextBox während der Laufzeit verändert werden. Längere Zeilen können in VB 2005 übrigens durch Anfügen eines von einem Unterstrich gefolgten Leerzeichens („ _“) umgebrochen werden (siehe Befehl zum Zeichnen der Pixel).

Es folgt die bekannte Hopalong-Formel und anschließend eine Zeile, die ich mir für eine Änderung der Farben während der Laufzeit ausgedacht habe: Der Ausdruck „Math.Abs(x – xx)“ macht eine Aussage drüber, wie stark ein beim Zeichnen erfolgender Sprung ist: Wenn man beobachtet, wie der imaginäre „Stift“ beim Zeichnen oft plötzlich nach innen oder außen springt, erscheint es logisch, in diesem Moment auch einen Wechsel der Farbe durchzuführen: Die Variable „counter“ wird bei Erfüllung der im Listing enthaltenen Bedingung um 1 erhöht und bei 255 wieder auf 1 gesetzt. Setzt man die Variable „counter“ nun an Stelle der RGB-Werte in den Befehl zum Zeichnen der Pixel ein, so ergeben sich die im Bild gezeigten Farben.

Durch das Subtrahieren der Variablen von 255 oder das Addieren von counter/2 zum Wert 127 ergeben sich begrenzte Möglichkeiten zum Zeichnen von Farben. Der Ausdruck „FromArgb (counter, counter, counter)“ würde lediglich zu Grau- und Schwarzweißwerten führen.

Besser wäre natürlich das Arbeiten mit vorgegebenen Paletten. Dies würde jedoch das Programm zu umfangreich und unübersichtlich machen und den Rahmen dieser Publikation überschreiten.

Eine andere Verbesserung des Programms, die die Berechnungszeit wesentlich abkürzen würde, wäre eine Überprüfung auf bereits gezeichnete Punkte. Damit ließe sich verhindern, dass Punkte mehrmals überschrieben würden, was, so meine Erfahrung, sehr viel Zeit spart, da dies sehr häufig vorkommt. Dies lässt sich zum Beispiel durch ein zweidimensionales Array erreichen, das den Koordinaten der gezeichneten Punkte entspricht, und das jedes Mal, bevor ein Punkt gezeichnet wird, abgefragt werden kann, ob solch ein Punkt bereits existiert. Da das Zeichnen von Punkten sehr viel länger dauert als das Abfragen eines Array-Inhaltes, gewinnt man damit in der Tat erheblich viel Zeit. Auch in diesem Falle möchte ich jedoch auf den betreffenden Code verzichten, um das vorliegende Programm nicht zu unübersichtlich werden zu lassen.

Und hier nun der lauffähige, kommentierte Programmcode:

```
Public Class Form1

    ' ########################################  PROGRAMM "HOPALONG" ZUM ZEICHNEN VON ORBIT FRAKTALEN
    ' ########################################  NACH DEM BARRY-MARTIN-ALGORITHMUS

    ' ########################################  Variablendeklaration  ##########################################

    Dim schleifenzahl As Double ' Anzahl der Iterationen
    Dim a As Double             ' Hopalong-Parameter
    Dim b As Double             ' Hopalong-Parameter
    Dim c As Double             ' Hopalong-Parameter
    Dim x As Double             ' Hopalong-Parameter
    Dim y As Double             ' Hopalong-Parameter
    Dim i As Double             ' Iterationsvariable
    Dim yy As Double            ' Hopalong-Parameter
    Dim xx As Double            ' Hopalong-Parameter
    Dim teilfaktor As Double    ' Größe der Abbildung
    Dim counter As Integer      ' Farb-Berechnung
    Dim jumpwert As Double      ' Farb-Berechnung
```

```
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

        ' Textboxen werden beim Start automatisch mit vorgegebenen Standardparametern gefüllt
        TextBox1.Text = 100000
        TextBox2.Text = 98
        TextBox3.Text = "0.44"
        TextBox4.Text = "23.234"
        TextBox5.Text = 100
        TextBox6.Text = 10

    End Sub
```

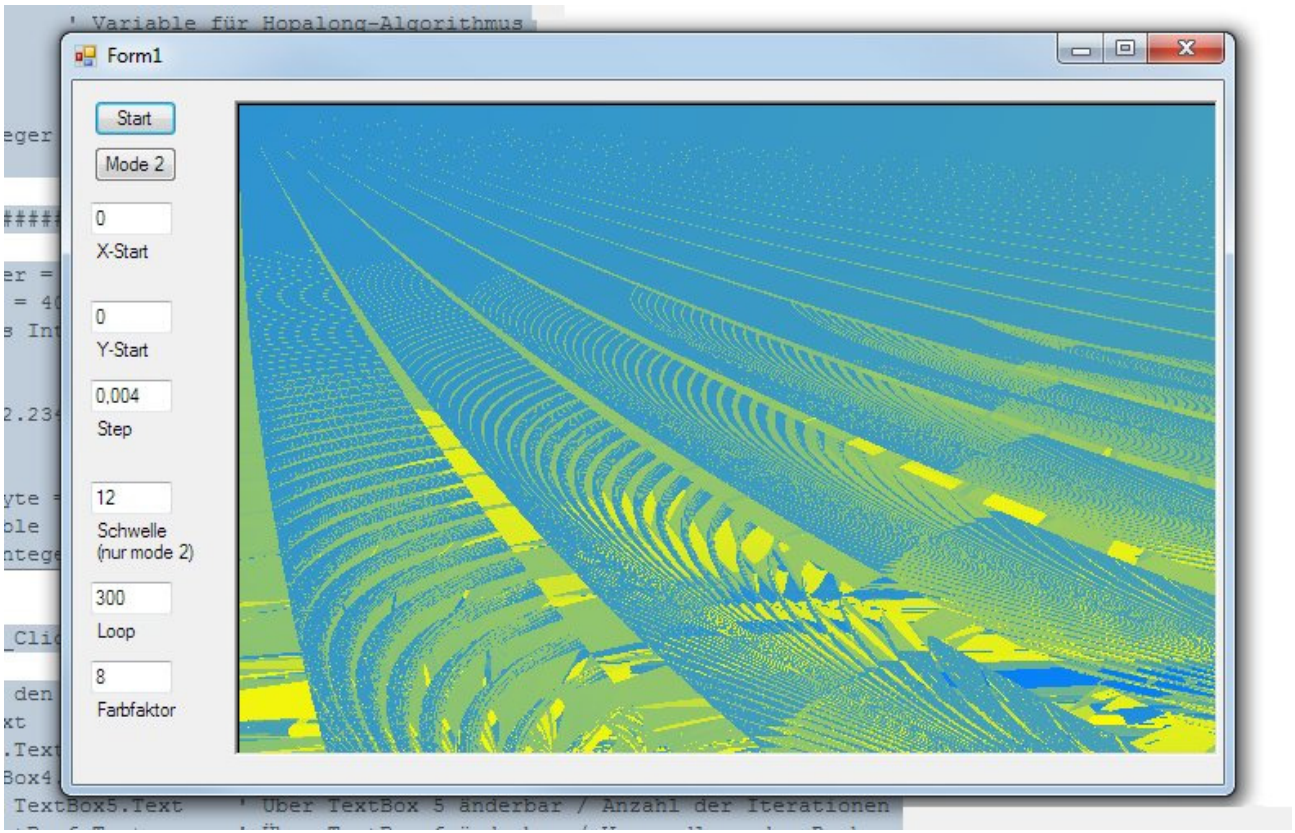
```
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

        'Reset verschiedener Hopalong-Parameter

        xx = 0
```

```
yy = 0
x = 0
y = 0
counter = 0

PictureBox1.CreateGraphics.Clear(Color.Black) 'Bildschirm löschen

'Den Variablen werden die in den TextBoxen gespeicherten Werte zugeordnet.
'Die Werte in den TextBoxen können vom Anwender vor dem Start einer neuen Grafik geändert werden
a = CInt(TextBox2.Text)
b = CInt(TextBox3.Text)
c = CInt(TextBox4.Text)
schleifenzahl = TextBox1.Text
jumpwert = TextBox6.Text
teilfaktor = TextBox5.Text

' Schleife zum Zeichnen der Grafik

For i = 1 To schleifenzahl '(Schleifenzahl kann bis zu mehreren Millionen betragen)

    If counter > 255 Then counter = 1 'Die Variable „counter“ sorgt für die Änderung der Farben beim Zeichnen

    ' Es folgt der leider etwas kryptische VB-Befehl zum Zeichnen der Pixel an den berechneten Stellen:
    ' In VB gibt es keinen Befehl zum Zeichen von Pixeln. Am besten eignet sich der Befehl zum Zeichnen von
    ' Rechtecken mit der Kantenlänge 1 (Fill Rectangle)
    ' Die Konstante mit dem Wert 300 bestimmt, wieweit die Grafik vom Nullpunkt (links oben)
    ' entfernt ist und kann individuell angepasst werden.

    PictureBox1.CreateGraphics.FillRectangle(New SolidBrush(System.Drawing.Color.FromArgb(255 - counter, _
    counter, 127 + counter / 2)), New Rectangle(10 * x / teilfaktor + 300, 10 * y / teilfaktor + 300, 1, 1))

    'Die Hopalong-Formel:
    xx = y - Math.Sign(x) * Math.Sqrt(Math.Abs(b * x - c))

    'Ein kleiner "Trick" zum sprunghaften Ändern der Farben
    If Math.Abs(x - xx) < jumpwert Then
        counter = counter + 1
    End If

    'Weiterer Teil der Hopalong-Formel
```

```
            yy = a - x

            'Dies sind Alternativen zur obigen Zeile, die ebenfalls interessante Ergebnisse liefern:
            'yy = b - x
            'yy = a + b - x
            'yy = a - b - x
            'yy = b * b - x
            'yy = Math.Sqrt(a * b) - x

            'Weiterer Teil der Hopalong-Formel
            x = xx
            y = yy

        Next

    End Sub
```

```
End Class
```

2) Hopalong-Special



Programmbeschreibung

Zum Verständnis dieses Programmes möchte ich auf die detaillierte Beschreibung des vorhergehenden Programms verweisen, da das vorliegende Programm darauf basiert. Im Folgenden werden daher hauptsächlich die Unterschiede dieses Programms zum Vorhergehenden beschrieben.

Das Programm „Hopalong-Special“ enthält die gleiche Iterationsschleife wie das „normale“ Hopalong-Programm. Diese Schleife ist jedoch hier in zwei weitere, verschachtelte Schleifen eingebettet. Im zuvor beschriebenen Programm wurden die durch den Hopalong-Algorithmus berechneten Werte als Koordinaten zum Zeichnen eines Pixels herangezogen. Im hier betrachteten Fall sind die beiden zusätzlichen äußeren Schleifen jedoch im Zusammenhang mit dem inneren Befehl zum Zeichnen von Pixeln so programmiert, dass sie zu einem zeilenweisen Beschreiben des Bildschirms führen, was zum Beispiel zu einer Bildfläche von 800 mal 600 Pixeln führen könnte.

Gleichzeitig dazu werden zwei der drei Hopalong-Parameter um einen konstanten, vorgegebenen Wert erhöht – zum Beispiel um den Wert 0,004. Nach jeder Erhöhung erfolgt ein erneuter Start der vom vorhergehenden Programm bekannten Hopalong-Iteration, wobei jedoch die Ergebnisse hier nicht zu einer Figur, sondern zu einem farbigen Pixel führen, das genau an der durch die beiden äußeren Schleifen festgelegten Stelle gezeichnet wird. Dabei ist es wichtig, beim Start jeder neuen Zeile den Wert des horizontal-inkrementierten Wertes wieder auf Null zu setzen, und den Wert des vertikalen Parameters erst beim Beginn einer neuen Zeile zu inkrementieren. Auf diese Weise ergibt sich eine Zuordnung der in den vier Quadranten der Zahlenebene existierenden Koordinatenpaare zu den Parametern: Das Verhalten des Hopalong-Algorithmus wird damit, ähnlich wie bei der Mandelbrotfigur, in Abhängigkeit zwei seiner Parameter beschrieben.

Natürlich ist diese Zuordnung willkürlich, aber sie führt zu einer klar definierten, reproduzierbaren Struktur, die aus seltsamen, völlig unregelmäßig angeordneten und jeder Symmetrie entbehrenden Objekten führt, in die man, wie bei der Mandelbrotfigur, hineinzoomen kann. Damit jedoch nicht bloß eine homogene Fläche gezeichnet wird, muss jedes einzelne Pixel beim Zeichnen noch einen bestimmten Farbwert erhalten, der im Zusammenhang mit den Eigenschaften der aktuellen Hopalong-Figur steht.

Bleibt die Frage, wie diese Eigenschaften in farbige Punkte verwandelt werden. Dazu gibt es viele verschiedene Methoden, die jedoch zu ähnlichen Ergebnissen führen: Zum Beispiel das Berechnen des Maximums innerhalb einer vorgegebenen Iterationszahl oder die „Messung“ der Schleifendurchläufe, die zum Überschreiten einer vorgegebenen Schwelle benötigt wurden. Das folgende Listing ist lauffähig. Die in der Galerie gezeigten Bilder wurden jedoch mit einer Software erzeugt, die über eine Zugriffsmöglichkeit auf Farbpaletten verfügt. Da diese Methode jedoch den Rahmen dieses Beitrags überschreiten und zur Unübersichtlichkeit des Listings führen würde, habe ich mich auf die gezeigte, einfachere Methode beschränkt.

```
Public Class Form1

' ################################### HOPALONG SPECIAL  ######################################

    Dim yyy As Integer      ' Äußere Schleifenvariable zum Zeichnen
    Dim xxx As Integer      ' Innere Schleifenvariable zum Zeichnen
    Dim i As Integer        ' Innerste Schleifenvariable zur Iteration
    Dim buffer As Double    ' Speichervariable für Zeichenmode 2
    Dim xx As Double        ' Variable für Hopalong-Algorithmus
    Dim yy As Double        ' Variable für Hopalong-Algorithmus
    Dim x As Double         ' Variable für Hopalong-Algorithmus
    Dim y As Double         ' Variable für Hopalong-Algorithmus
    Dim farbwert As Integer ' Farb-Umrechnung
    Dim flag As Byte        ' Schaltvariable für Zeichenmode 2

    ' ################################## Parameter ###############################################

    Dim breite As Integer = 600   ' Bildbreite (Darf geändert werden)
    Dim höhe As Integer = 400     ' Bildhöhe   (Darf geändert werden)
    Dim innenschleife As Integer  ' Maximaler Iterationswert; Festgelegt in TextBox 5
    Dim a As Double               ' Hopalong-Parameter und gleichzeitig linker oberer Startpunkt (X) der Grafik
    Dim b As Double               ' Hopalong-Parameter und gleichzeitig linker oberer Startpunkt (Y) der Grafik
    Dim c As Double = 32.234      ' Dritter Parameter für die Hopalong-Gleichung. Darf verändert werden.
    '                               Hier können andere Werte jedoch zu völlig anderen Ergebnissen führen
    Dim stap As Double            ' Über Textbox 3 änderbar: Schrittweite bzw. Auflösung
    Dim graphmode As SByte = -1   ' Kann mit Button 2 in 1 geändert werden
    Dim schwelle As Double        ' Nur relevant für grafikmodus 2
    Dim farbfaktor As Integer     ' Kann geändert werden
```

```
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

        'Abfrage der in den TextBoxen gespeicherten, vorgegebenen Variablen
        a = TextBox1.Text               ' Über Textbox 1 änderbar / Startkoordinate X
        stap = TextBox3.Text            ' Über TextBox 3 änderbar / Auflösung
        schwelle = TextBox4.Text        ' Über TextBox 4 änderbar / Schwelle für mode 2
        innenschleife = TextBox5.Text   ' Über TextBox 5 änderbar / Anzahl der Iterationen
        farbfaktor = TextBox6.Text      ' Über TextBox 6 änderbar / Verwandlung der Rechenergebnisse in Farben
```

```
For yyy = 1 To breite ' Erste, äußere Verschachtelungsebene: Bildbreite: X-Koordinate

 a = a + stap 'Inkrementierung des Parameters a um den Wert stap (z.B. 0,004)
 b = TextBox2.Text 'Neue "Zeile": b muss auf Anfangswert zurückspringen und wird neu incrementiert

 For xxx = 1 To höhe ' Zweite Verschachtelungsebene: Bildhöhe: ’ Y-Koordinate zum Zeichnen

     b = b + stap 'Inkrementierung des Parameters b um den Wert stap

     buffer = 0   ' Farbwertberechnung: Buffer-Reset
     xx = 0       ' Hoplaong-Parameter Reset
     yy = 0       ' Hoplaong-Parameter Reset
     x = 0        ' Hoplaong-Parameter Reset
     y = 0        ' Hoplaong-Parameter Reset

     For i = 1 To innenschleife  ’Dritte, innere Verschachtelungsebene
         ' Hoplaong-Schleife - Kann von ca. 100 bis unbegrenzt variiert werden
         ' Hopalong-Gleichung:
         xx = y - Math.Sign(x) * Math.Sqrt(Math.Abs(b * x - c))
         yy = a - x
         ' Berechnung der Farbe des zu zeichnenden Pixels
         ' Zwei Methoden zur Farbberechnung: Graphmode 0 und 1. Der Modus kann von der
         ' Benutzeroberfläche aus gewählt werden.

         ' Mode 1: Speichern des höchsten Wertes in „buffer“
         If graphmode = -1 Then
             If x > buffer Then buffer = Math.Abs(x)
         End If

         ' Mode 2: Abbruch der Schleife bei Erreichen eines Schwellenwertes

         If graphmode = 1 Then
             flag = 1
             If Math.Abs(x) > schwelle Then
                 If flag = 1 Then
                     buffer = i  'Schleifenwert i wird in Buffer gespeichert
                     Exit For 'Schleife wird verlassen
                 End If
             End If
```

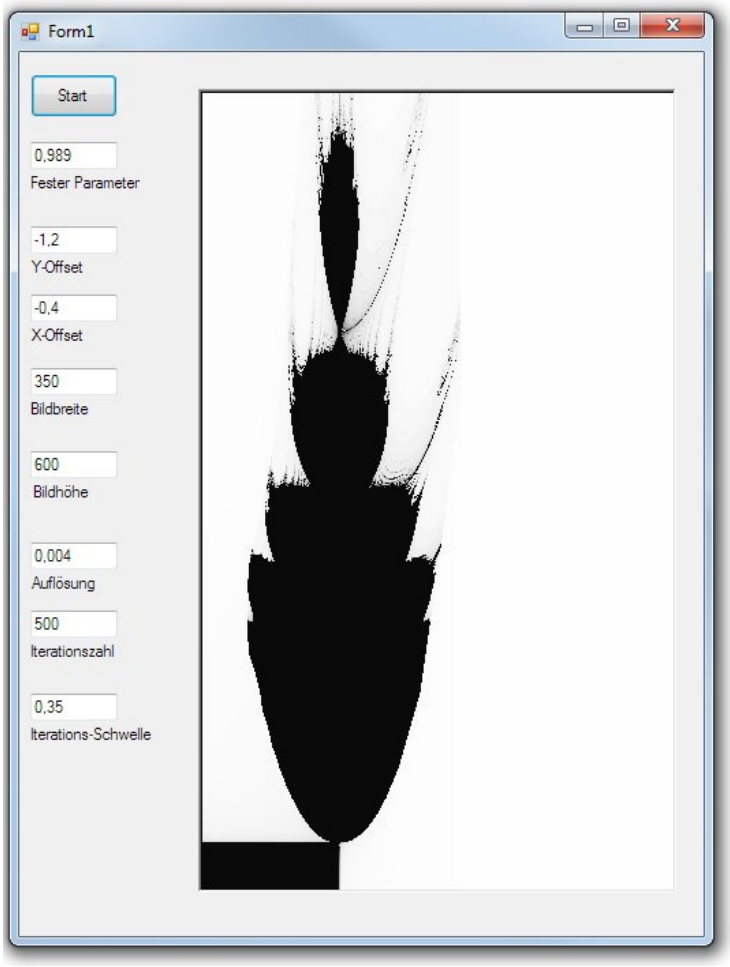
```
                    flag = 0
                End If
                ' Teil der Hopalong-Gleichung:
                x = xx
                y = yy

            Next i 'Ende der inneren Schleife zur Berechnung eines einzigen Punktes

            ' Umrechnung in Farbwert. Variable "farbfaktor" kann in TextBox 6 geändert werden
            farbwert = CInt(farbfaktor * buffer)

            If farbwert > 511 Then farbwert = 511
            If farbwert > 255 And farbwert < 512 Then farbwert = farbwert - 256

            ' Es folgt das Zeichnen des betreffenden Farbpixels an den Koordinaten yyy und xxx.
            ' Der Einsatz der Variablen "farbwert" als rgb-Werte ist nur ein Beispiel.
            ' Die besten Ergebnisse lassen sich mit Paletten-Arrays erzielen, in denen zu jedem Wert
            ' der Variablen "farbwert" die jeweiligen RGB-Werte gezielt nach Gesichtspunkten der
            ' optimalen Wahrnehmung bzw. der Ästhetik eingetragen sind.
            PictureBox1.CreateGraphics.FillRectangle(New SolidBrush _
            (System.Drawing.Color.FromArgb(farbwert, 125 + farbwert / 2, 255 - farbwert)), _
            New Rectangle(yyy, xxx, 1, 1))

        Next xxx
    Next yyy

End Sub
```

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    'Beim Start des Programms werden die Textfelder mit vorgegebenen Standardwerten gefüllt.
    'Diese Werte können vor einem Start mit dem Startbutton, so lange das Programm nicht mit Zeichnen
    ' beschäftigt ist, verändert werden.

    TextBox1.Text = 0
    TextBox2.Text = 0
    TextBox3.Text = "0,004" 'Kommazahlen funktionieren nur, wenn sie als Strings vorgegeben wurden
    TextBox4.Text = 12
    TextBox5.Text = 300
```

```
        TextBox6.Text = 8
        Button2.Text = "Mode 2"
    End Sub
```

```
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
        'Dieser Button schaltet abwechselnd zwischen den beiden modi 1 und 2 zum Zeichnen um (Toggle-Betrieb).
        graphmode = -graphmode
        If graphmode = -1 Then Button2.Text = "Mode 2"
        If graphmode = 1 Then Button2.Text = "Mode 1"
    End Sub
End Class
```

3) Mira-Special



Das Programm funktioniert ähnlich wie das vorhergehende (Hopalong-Special), mit dem Unterschied, dass sich im Zentrum der Schleife hier nicht der Hopalong-, sondern der Mira-Iterationsalgorithmus befindet. Weitere Erläuterungen finden Sie in den Kommentaren des (lauffähigen) Listings. Auch dieses Programm ist nicht identisch mit demjenigen, das die in der Galerie gezeigten Bilder erzeugt hat, da es bei der Farbgebung aus Gründen der Übersichtlichkeit auf eine Palette verzichtet.

Weitere Informationen zum Mira-Algorithmus finden Sie an anderer Stelle in der Fraktalwelt sowie auf zahlreichen weiteren Webseiten.

```
Public Class Form1

    ' ##################################### Variablendeklaration ######################################

    Dim b As Double           ' mira-Parameter
    Dim a As Double           ' mira-Parameter
    Dim c As Double           ' mira-Parameter
    Dim z As Double           ' mira-Parameter
    Dim x As Double           ' mira-Parameter
    Dim j As Double           ' mira-Parameter
    Dim y As Double           ' mira-Parameter

    Dim xx As Integer         ' Koordinaten zum Zeichnen
    Dim yy As Integer         ' Koordinaten zum Zeichnen
    Dim schritt As Double     ' X-Auflösung
    Dim schritt2 As Double    ' Y-Auflösung
    Dim grenzwertx As Integer ' Bildbreite
    Dim grenzwerty As Integer ' Bildhöhe
    Dim wert As Integer       ' Zeichenfarbe
    Dim stap As Double        ' Gemeinsame X-Y-Auflösung
    Dim i As Integer          ' Iterationsvariable
    Dim innenzahl As Integer  ' Anzahl der Iterationen
    Dim schwelle As Double    ' Iterationsschwelle

    'Dim xp As Double          Nicht verwendet
    'Dim yp As Double          Nicht verwendet
    'Dim zoom As Integer       Nicht verwendet


    ' ##################################### MIRA SPECIAL ##################################################

    ' Dieses Programm interpretiert zwei von drei Mira-Paramtern als X- bzw. Y-Koordinaten
    ' und zeichnet das Verhalten des Mira-Attraktors als Farben bzw. Helligkeitswerte in das
    ' entsprechende Achsensystem. Bei vorgegebener Wahl der Parameter ergibt sich ein fraktales
    ' Gebilde, das Ähnlichkeit mit dem Apfelmännchen besitzt und in das hinein gezoomt werden kann.
    '
    ' Die Möglichkeiten dieses Programms können jedoch erst voll ausgeschöpft werden, wenn der Variablen
    ' "wert" ein RGB-Farbarray (Palette) zugeordnet wird.
```

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles   Button1.Click

        'Reset der beiden Variablen:
        schritt = 0
        schritt2 = 0

        PictureBox1.CreateGraphics.Clear(Color.Black) 'Bildschirm löschen
        ' Den Variablen werden die voreingestellten Werte aus den Picture-Boxen zugeordnet
        grenzwertx = TextBox4.Text
        grenzwerty = TextBox6.Text
        stap = TextBox7.Text
        innenzahl = TextBox8.Text
        schwelle = TextBox10.Text

        b = TextBox1.Text 'Der fixen Variablen b wird der in TextBox1 stehende Wert zugeordnet

        For yy = 1 To grenzwerty 'Äußere Schleife (vertikaler-Fortschritt)

            a = TextBox2.Text + schritt2 ' Inkrementierung  der mirar-Konstanten um "schritt2"

            For xx = 1 To grenzwertx 'Schleife zum horizontalen Fortschritt. XX wird um 1 incrementiert

                y = TextBox3.Text + schritt 'incrementierung der mira-Konstanten um den Wert "schritt"
                'Reset bestimmter mira-Konstanten
                x = 0
                j = 0
                z = 0
                'xp = 0
                'yp = 0
                c = 2 - 2 * a

                For i = 1 To innenzahl 'mira-Iterationsschleife
                    z = x
                    x = b * y + j
                    j = a * x + c * (x ^ 2) / (1 + x ^ 2)
                    y = j - z
                    'xp = zoom * x + 400 'Koord. zum Zeichnen der mira-Figur, hier nicht benötigt
                    'yp = zoom * y + 400 'Koord. zum Zeichnen der mira-Figur, hier nicht benötigt
```

```
                If x > schwelle Then Exit For

                'Abbruch bei Überschreiten des vorgegebenen Wertes "schwelle"

              Next i

            wert = i 'Zuordnung der benötigten Schleifendurchläufe zur Farb-Variablen "wert"

            'wert darf nicht kleine 0 und nicht größer 255 werden
            If wert > 255 And wert < 512 Then wert = wert - 256
            If wert > 511 Then wert = 255
            If wert < 0 Then wert = 0

            'Zeichnen der Pixel in die PictureBox
            PictureBox1.CreateGraphics.FillRectangle(New SolidBrush(System.Drawing.Color.FromArgb(255 -
             wert, 255 - wert, 255 - wert)),New Rectangle(xx, yy, 1, 1))

            schritt = schritt + stap ' incrementierung der Variablen schritt

        Next xx

        schritt = 0 'Zeile "springt nach links an den Anfang zurück
        schritt2 = schritt2 + stap 'Icrementierung des vertikalen Zeilenvorschubs
    Next yy

End Sub

Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

    'Füllen der TextBox-Felder mit den voreingestellten Werten
    'geschieht automatisch beim Programmstart
    TextBox1.Text = "0,989" 'Parameter
    TextBox2.Text = "-1,2"
    TextBox3.Text = "-0,4"
    TextBox4.Text = 350 'Bildbreite x
    TextBox6.Text = 600 'Bildhöhe y
    TextBox7.Text = "0,004" 'Schrittweite stap
    TextBox8.Text = 500 'Iterations-Schleifenzahl innen
    TextBox10.Text = "0,35"
End Sub
```

```
Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles _
Button2.Click

       End 'Beenden des Programms

End Sub

End Class
```