Algorithmus „Region Growing“

```
Eingabe:
            I : DRI
            I_pnv : positive Normalenvektoren zu I
            t : Grad der Polynome
            T : Abstandstoleranz

Ausgabe:
            {R_q} : Regionen
            {z_q} : zugehörige Polynome

lokale Var.:
              n : Anzahl der Regionen
              V : Toleranz des Normalenvektors
              H : Schwellwert
              RegionGrows : Flag
              RemainingQuads : verbleibende Vierecke
              D_q : min./max. Zeilen- und Spaltennummer (4 Werte)

begin
  n = 0; RemainingQuads = Gesamtzahl der Vierecke in I;

  while RemainingQuads > 0 do
          n := n + 1;
          SelectSeedRegion(I, I_pnv, R_n, D_n);          /* schließt Interaktion ein */
          LeastSqPolynomialFitting(R_n, t, z_n);
          ThresholdSelection(R_n, z_n, H);          /* schließt u. U. Interaktion ein */
          RegionGrows := true;

          while ((RegionGrows == true) or (H < T)) do

                  LocalRegionGrowing(I, R_n, D_n, z_n, t, H, V, T, RegionGrows);

          done

          RemainingQuads := RemainingQuads − |R_n|;
  done
  /* jetzt ist n = N, also die Gesamtanzahl der ermittelten Regionen */
end
```

Algorithmus „LocalRegionGrowing“

Eingabe:
I : DRI
t : Grad der Polynome
T : Abstandstoleranz

Ausgabe:
$RegionGrows$: Flag

Ein-/Ausgabe:
$\{R_n\}$: Regionen
D_n : min./max. Zeilen- und Spaltennummer (4 Werte)
$\{z_n\}$: zugehörige Polynome
H : Schwellwert
V : Toleranz des Normalenvektors

lokale Var.:
n : Anzahl der Regionen
$OldRegionQuads$: verbleibende Vierecke
$GrowOut$: Flag: Herauswachsen aus akt. Domain erlauben?
$SegmentQuads$: optimale Anzahl von Vierecken für akt. Region
$OutQuads$: Anzahl zu z_n inkompatibler Vierecke in akt. Region
$TotalQuads$: Anzahl kompatibler Vierecke
$RegionQuads$: Anzahl Vierecke in R_n
Lt, Rt, Up, Dn : Flags für das Wachsen in best. Richtung

begin
$GrowRegionIN(I, R_n, D_n, z_n, H, V, RegionQuads, OutQuads,$
$OldRegionQuads, TotalQuads, D_n, Lt, Rt, Up, Dn);$

/* *„inkompatible“ Vierecke werden vorr. nicht verwendet* */
$SegmentQuads := TotalQuads - OutQuads;$

if $(H > C_{large} * T)$ **then** /* *Verhindern der Entartung der Fläche* */
$GrowOut :=$ **false**;
elseif $(RegionQuads > C_{full} * SegmentQuads)$ **then**
/* *Region ausreichend gut gefüllt?* */
$GrowOut :=$ **true**;
elseif $(RegionQuads - OldRegionQuads > C_{in} * SegmentQuads)$ **then**
/* *Region innen ausreichend gewachsen?* */

```
      GrowOut := false;
   elseif (RegionQuads < C_gap * SegmentQuads) then
                                               /* noch zu viele Löcher innen? */
      begin
         H erhöhen; GrowOut := false;
      end
   else
      GrowOut := true;
   fi

   if (GrowOut == true) then                  /* Herauswachsen erwünscht? */
      GrowRegionOUT(I, D_n, z_n, H, V, R_n, D_n, N_gr, RegionQuads, Lt, Rt, Up, Dn);
   fi

   if (RegionQuads > C_grow * OldRegionQuads) then
                                               /* Wuchs die Region ausreichend? */
      RegionGrows := true;
   else
      begin
         RegionGrows := false;
         H erhöhen;                            /* Toleranz erhöhen */
      end
   fi

   if ((RegionGrows == true) or (H < T)) then
                          /* z_n nicht neu anpassen, wenn Prozeß abgeschlossen */
      LeastSqPolynomialFitting(R_n, t, z_n);
   fi

end
```