# Tartalomjegyzék

II

# Szín, textúra és mozgás alapú szegmentálás Gradient Vector Flow segítségével

Horváth Péter, Kató Zoltán

Szegedi Tudományegyetem, Informatikai Tanszékcsoport,
Pf. 652, 6701 Szeged, Fax: 62/546 397
Horvath.Peter.3@stud.u-szeged.hu, kato@inf.u-szeged.hu

**Absztrakt.** Az *aktív kontúr* azaz *snake* közkedvelt eszközök a képfeldolgozásban és szegmentálásban. Az aktív kontúr olyan görbe amelyekhez egy energiát tudunk rendelni, mely energia a görbe mozgatásával vátozik, így az energia minimalizálásával alkalmas élek detektálására, képek szegmentálására, felületek modellezésére (3D-ben). Két alapvetően különböző aktív kontúr létzik, paraméteres aktív kontúr (*parametric active contour*) és geometriai aktív kontúr (*geometric active contour*). A Snake energiáját két komponenes alkotja, a belső energia (*internal force*), amleyet a kontúr alakja befolyásol és a külő energia (*external force*), mely a kontúr környezetétől függ. Cikkünkben a *Gradient Vector Flow (GVF)* nevű eljárás kiterjesztéseit mutatjuk be ($\Re^2 \rightarrow \Re^n$). Az eredeti módszer szürkeárnyalatos képek szegmentálálására alkalmas, bemutatunk egy lehetőséget mely segítségével több dimenziós esetkben is használhatjuk a *GVF*-et, majd néhány példát *CIE L*u*v** színtérben, *Optical Flow* szegmentálására és textúrázott képek esetére. Bemutatunk egy technikai megoldást amely segítségével gyorsíthatunk az algoritmuson.

## 1. Bevezetés

Az Aktív kontúrok olyan görbék, melyek a kép területén mozognak abban az irányban, amerre az őket mozgató energiák mutatnak. Kass, Witkin és Terzopoulos 1987-ben megjelent cikkében elsők között [3] mutatják be a paraméteres aktív kontúr modellt. Terzopoulos további cikkeiben [6] [7] foglakozott alakmodellezéssel és a jelen cikk egyik fő részét képző mozgás követéssel. A tradicionális *snake* modellek problémája, hogy vannak olyan konkáv alakzatok, melyekre körülményes a használatuk. 1998-ban C. Xu és J. L. Prince publikálták a Gradient Vector Flow [8] előállításáról szóló cikküket. A módszer segítségével tetszőleges konkáv alakzatokra ráhúzódó snake készíthető. A GVF-et szürke árnyalatos képek szegnetálására használhatjuk, itt bemutataunk egy módszert melyet olyan általános esetekben tudunk használni, amikor több jellemző áll rendelkezésünkre (pl.: RGB, CIE L*u*v*, Optical Flow,...).

## 2. Aktív kontúrok

Ebben a fejezetben bemutatjuk a tradicionális paraméteres snake modellt és annak hibáit, ismertetjük a Gradinet Vector Flow-t majd a több dimenziós kiterjesztését. Végül egy olyan snake struktúrát mutatunk be, mely könnyen megvalósítható és gyors.

### 2..1. Paraméteres snake modell

A snake egy olyan paraméteres görbe, amely a következőképpen írható fel $\overline{x(s)} = [x(s), y(s)], s \in [0, 1]$, ezen függvény energiáját a következőképpen definiáljuk:

$$\int_0^1 \frac{1}{2}\left(\alpha|x'(s)|^2 + \beta|x''(s)|^2\right) + E_{ext}(x(s))ds, \tag{1}$$

ahol $\alpha$ és $\beta$ súlyparaméterek, az $x'(s)$ és $x''(s)$ az első és második derivált a kontúr feszességét és merevségét jellemzi. A külő energiát a következőképpen definiálták:

$$E_{ext}(x, y) = -|\nabla I(x, y)|^2, \tag{2}$$

azaz a kép gradienseinek értékei adják az energiát. Az energiafüggvény mininalizásáról bővebben [8]-ben olvashatunk.

**Problémák a klasszikus aktív kontúrral** A snake egy vektormező mentén mozog, ezt a vektormezőt kell megkonstuálnunk. Erre a klasszikus megoldás egy távolság definícióval minden pontból egy alacsonyabb energiájú rész felé mutató vektor, a távolság lehet Euklideszi vagy Chamfer definíciója alapján [1]. A probléma ezekkel a módszerekkel, hogy az alakzatok egyes részein nem megfelelően használható a vektormező. Erre a problémára találhatunk megoldást a [8]-es cikkeben.

### 2..2. Gradient Vector Folw

Egy iteratív módszer, mely a konkáv problémák esetében segít a jó megoldást megtalálni. Egy adott $(x, y)$ pontban a vektor irányát a következőképpen kaphatjuk meg:

$$\overrightarrow{x}_t(s, t) = \alpha x''(s, t) - \beta x''''(s, t) + \overrightarrow{v}, \tag{3}$$

ahol $v(x, y) = E_{ext}$ a külső energia.

**Gradient Vector Flow** C. Xu és J. L. Prince cikkükben [8] definiáltak egy éltérképet, legyen $f(x, y)$ az $I(x, y)$ éle, amelyet például a Sobel operátorok segítségével könnyedén kiszámolhatunk

$$f(x, y) = \sqrt{s_1(i, j)^2 + s_2(i, j)^2}. \tag{4}$$

A GVF egy $\overrightarrow{v}(x, y) = [u(x, y), v(x, y)]$, amely vektorok minimalizálják a következő egyenletet:

$$\varepsilon = \int\int \mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) + |\nabla f|^2 |\overrightarrow{v} - \nabla f|^2 dxdy \tag{5}$$

A (1 a, b és c) ábrán GVF segítségével előállított képek láthatók, de az a ábrán látható körvonal a 2..3.. fejezetben ismertetett módszerrel készült, azaz új pontok beszúrásával és felesleges pontok elhagyásával.

### 2..3. Pontok felvétele és elhagyása a GFV snake-ből

Az itt bemutatott gyakorlati megvalósítás inicialzálásakor felveszünk pontokat, melyeknek nem kell szomszédos pixelen lenniük (ezek alkotják), de egy listában vannak tárolva így a sorrendjük szigorúan rögzített. Ezeket a pontokat fogjuk a GVF által kreált vektormezőn mozgatni.

**Új pontok felvétele** Amikor inicializáltuk a snake-et és elkezdjük iteratív lépésekkel a GVF által megatározott irányába mozgatni előfrodulhat, hogy két pont nagyon eltávolodik egymástól és ekkor a közöttük lévő területre nem jut olyan pont amely ráhúzódhatna a korrekt pozícióra. Ez főleg "U" alakú részeken és szakadások közelében figyelhető meg (1 e). Ekkor beszúrunk egy újabb pontot a kontúrba, úgy, hogy az a két eltávolodott pontot összekötő egyenesen a pontok között középen legyen. Ezzel nem rontunk a snake állapotán, és az új pont elindulhat a konkáv részek irányába. A módszer egy lehetséges implemetálása, ha megkeressük a $k$ legnagyobb távolságot a snake mentén és oda beszúrunk egy-egy pontot. Tapasztalataink szerint egy $n$ pontot tartalmazó kontúrnál $k = n/10 \dots n/100$ választása a legjobb. A (1 e) ábrán láthaó a beszúrás nélküli eredmény. Megfigyelhető, hogy a szakadásba nem jutott pont és a konkáv, de "U" alakú területekre nem konvergált be a snake.

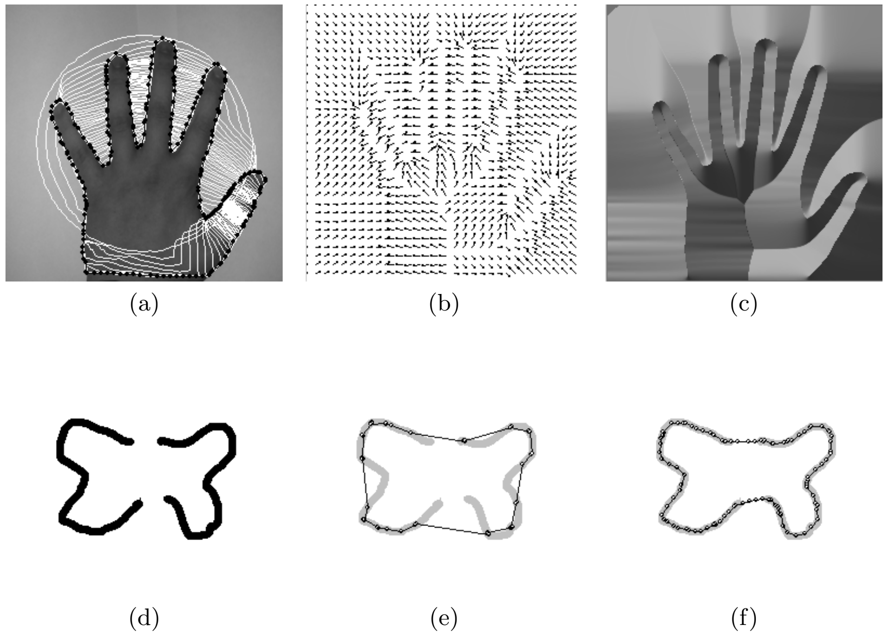
**Pontok törlése** Ha minden iterációban beszúrunk $k$ pontot, akkor az $i$. iterációban már $n+ik$ db pontunk lesz, mely nem rontja el a snake jellemzőit, de a számítási időt jelentősen befolyásolhatja. Ezért érdemes minden iterációban kitörölni a felesleges pontokat. Ezek olyan pontok amelyek túl közel esnek egymáshoz. Azaz minden $(p_{i-1}, p_{i+1})$ párosra vizsgáljuk meg milyen közel esnek egymáshoz, és a $k$ legkisebbre töröljük ki a $p_i$ pontot.

### 2..4. Gradient Vector Flow kiterjesztése több dimenziós jellemzővektorokra

Az eddig ismertetett módszer $\Re^2 \to \Re$-be történő leképezésekre, azaz szürkeárnyalatos képekre lett definiálva. Az optical flow például $\Re^2 \to \Re^2$ képező függvény, de az RGB vagy a CIE L*u*v* $\Re^2 \to \Re^3$, azaz a GVF-et nem tudjuk közvetlenül fehasználni, ezért általánosítjuktöbb dimenzióra. Feladatunk a következő: adott $f(x,y) : \Re^2 \to \Re^n$ és ehhez készítsük el a GVF-et. A szürkeárnyalatos kép [1] parciális deriváltjvésszük, ez több dimenziós függvények esetében ez egy Jacobbi mátrix, és a következőképpen írható fel:

$$Jac(f(x,y)) = \begin{pmatrix} \frac{\partial f_1(x,y)}{dx} & \dots & \frac{\partial f_n(x,y)}{dx} \\ \frac{\partial f_1(x,y)}{dy} & \dots & \frac{\partial f_n(x,y)}{dy} \end{pmatrix}_{2\times n}, \tag{6}$$

[1] A szürkeárnyalatos képet most tekintsük egy $f : \Re^2 \to \Re$ függvénynek.




**1. ábra:** Gradient Vector Flow seítségével előálított képek. Az a ábrán egy kézfejre ráhúzodott snake, b-n a vektormező és a c-n a vektorok színskálán ábrázolva. Konkáv alakzat szakadással látható a d ábrń, e ábra a GVF snake beszúrás és törlés nélkül ($n = 200$), f ábrán beszúrás és törléses GVF snake ($n = 200, k = 2$).

legyen $M \in \Re^{(2 \times n)}$, és $M = Jac(f(x, y))$. Bontsuk fel az $M$-et 2 sorvektorára, ezek legyenek $M_x, M_y \in \Re^n$ :

$$M_x = (M_{1,1}, \ldots, M_{1,n}), M_y = (M_{2,1}, \ldots, M_{2,n}), \tag{7}$$

így a parciális deriváltak koordinátánként a két vektorban vannak. Ezután mivel lehetséges, hogy bizonyos jellemzőket nem ugyanakkora súllyal akarunk figyelembe venni, súlyozzuk egy $\alpha$ súlyvektorral ezeket, legyen $\alpha \in \Re^n$. Ebből a GVF-nél használt $f_x(x, y), f_y(x, y)$ iránykomponenseket úgy kapjuk meg, hogy vesszük az $M'_x, M'_y$ vektorok normáit, azaz:

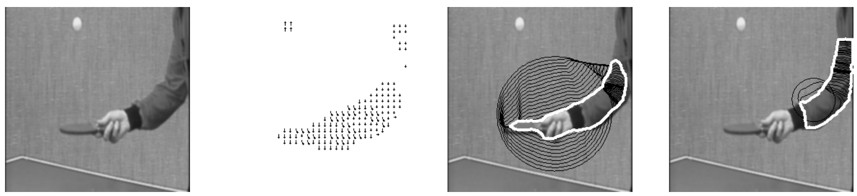
$$f_x(x, y) = \|M_x\|_k, f_y(x, y) = \|M_y\|_k, k = 1, 2, \infty, max. \tag{8}$$

## 3. Kísérleti eredmények

A kísérletek során különböző több dimenziós jellemzőket vizsgáltuk, ezek:

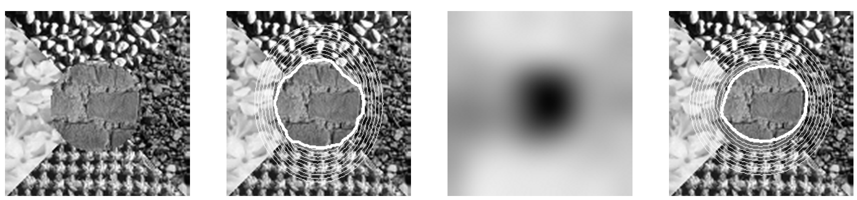
- optical flow
- szín (CIE L*u*v* színtérben [5])
- textúra *Gabor-filter* [2] és *MrSAR* (Multiresolution Autoregressive modell) [4] szűrők felhasználásásval

**Szegmentálás mozgás alapján** A (2 a). ábrán egy asztaliteniszező játékos látható, mely egy képsorozat egy szekvenciája, a (2 b). ábrán a kép és a rákövetkező kép közötti mozgások vektorai láthatók. Végül a szegmentálás eredménye a (2 c). ábrán. A Gradient Vector Flow paraméterei: $n = 200, k = 3$. A kontúrt inicializáláskor kör alakú volt, és a játékos kezét akartuk szegmentálni az optical flow segítségével, igy a két jellemző az optical flow vektorainak $x$ és $y$ koordinátái az $\alpha$ súlyvektor komponensei egyenlők voltak. A (2 c) ábrán látható a snake mozgása, a fekete kontúrokat minden 5. iteráció után rajzotuk ki. A snake kb. 100 iteráció után bekovergált.



(a) (b) (c) (a)

**2. ábra:** Kép szegmentálása mozgás alapján, az a ábrán az eredeti kép látható a b-n az optical flow, a c-n pedig a szegmentálás eredménye, és a kontúr mozgása, a d ábrán a játékos karjának szegmentálása szín alapján.



(a) (b) (c) (d)

**3. ábra:** Kép szegmentálása textúra alapján, az a ábrán láthatjuk az eredeti képet, a b-n az MrSAR algoritmussal kinyert jellemzők alapján szegmentált képet, a c ábrán a Gabor-filter által kinyert jellemzők a d-n pedig az ezek segítségével készített szegmentálás látható.

**Textúra alapú szegmentálás** A (3 a és b). ábrákon textúra alapú szegmentálást láthatunk. Az eredeti kép különböő rendezett és véltelen textúrák felhsználásával szintetikusan lett előállítva. A textúrázott képből MrSAR algoritmus segítségével nyertk ki a textúrajellemzőket. Az MrSAR algoritmus főleg a véletlen textúrák feldolgozásakor használható eredményesen, ha a képen rendezett textúrák vannak akkor Gabor-filter használata előnyösebb. A (3 a) képen a középen található kör alapú textúra részt akartuk szegmentálni, a (3 b) ábrán láthatjuk a szegmentálás eredményét és a kontúr mozgását, a snake alakja inicializáláskor ellipszis volt. A (3 c) ábrán Gabor-filter segístégével kinyert jellemzőket láthatjuk, a (3 d) ábrán pedig a Gabor-filterek felhasználásával szegmentált képet láthatjuk. Mivel szintetizált képet szegmentálunk a hibát könnyedén tudjuk mérni, a következő táblázat tartalmazza a kontúr átlagos átmérőjét, az pontok szórását és a futásidőt.

| Név | Sugár (r) | Szórás ($\sigma$) | Futásidő (ms) | Lefedett terület (százalék) |
|---|---|---|---|---|
| Eredeti | 64 | - | - | - |
| MrSAR | 59.8377 | 2.1756 | 6054 | 89.4 |
| Gabor-filter | 55.1819 | 3.5 | 6008 | 86.1 |

**Szegmentálás szín alapján** A (2 d) ábrán az asztaniteniszező karjának szegmentálását találjuk meg, látható, hogy nem sikerült az egész karra ráhúzódnia a kontúrnak, mert felül az alakzat konvex, azaz szélesedik az átmérője. A színek CIE L*u*v* színtérbe lettek konvertálva. A kontúr inicializálása egy körrel történt, a kör a játékos karján helyezkedik el.

## Irodalom


1. H. Borgeforst: Hierarchial Chamfer Matching: A Parametric Edge Matching Algorithm, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 10. No. 6., November 1988.
2. A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognition*, vol. 24, no. 12, pp. 1167–1186, 1991.
3. M. Kass, A. Witkin, D. Terzopoulos: Snakes: Active contour models, Int. J. Comput. Vis., vol. 1, pp. 321-331, 1987.
4. J. Mao and A. K. Jain, "Texture classification and segmentation using multiresolution simultaneous autoregressive models," *Pattern Recognition*, vol. 25, no. 2, pp. 173–188, 1992.
5. S. J. Sangwine and R. E. N. Horne, editors. *The Colour Image Processing Handbook.* Chapman & Hall, 1998.
6. D. Terzopoulos, K. Fleischer: "Deformable models," Visual Comput., vol. 4, pp. 306-331, 1988.
7. D. Terzopoulos, R. Szeliski: "Tracking with Kalman Snakes," in Active Vision, Cambridge, MA: MIT Press, 1992, pp. 3-20.
8. C. Xu and J. L. Prince: Snakes, Shapes, Gradient Vector Flow, IEEE Transactions on Image Processing, Vol. 7, no. 3, march 1998 pp. 359-369