علی آراسته – ۴۰۰۲۰۶۱۵۴

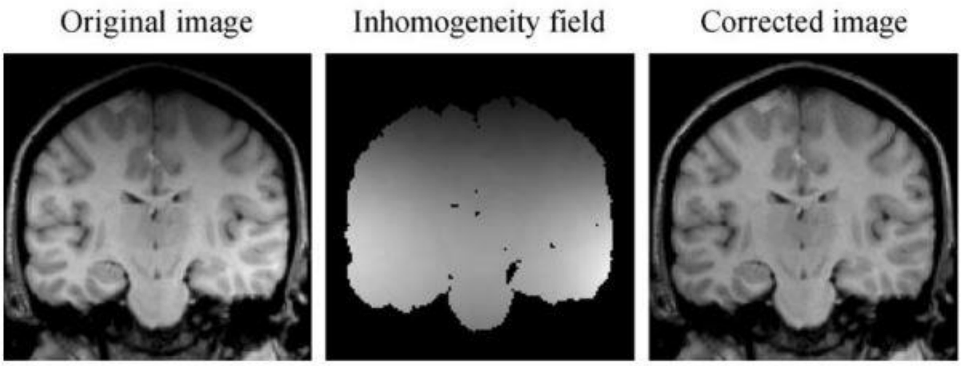
پردازش و تحلیل تصاویر پزشکی – تمرین سری پنجم

بخش تئوری

سوال ۱: Magnetic Resonance Imaging (MRI)، یا تصویربرداری رزونانس مغناطیسی، روشی است که با تحلیل فعل و انفعال بین بدن انسان و امواج رادیویی قرار گرفته بر روی یک میدان مغناطیسی قوی، منجر به تصاویر دارای جزئیات می‌شود. به دلیل رزولوشن مکانی و کنتراست تصویربرداری بالا، MRI به طور گسترده در تصویربرداری بالینی مورد استفاده قرار می‌گیرد؛ اما این نوع از تصویربرداری، به دلیل حضور تغییرات نرم شدت میدان در نواحی مختلف تصویر MR، با یک مشکل اساسی در تفسیر تصویر MR ساختاری، به صورت کمی و کیفی، همراه است. از این مشکل معمولا با عنوان Intensity Inhomogeneity یا ناهمگنی شدت میدان یاد می‌شود.

مولفه‌های اصلی تاثیرگذار در بزرگی و مشخصات مکانی ناهمگنی شدت میدان، قدرت میدان استاتیک، یکنواختی کاهش یافته سیم‌پیچ RF، میزان نفوذ RF، جریان‌های گردابی ایجاد شده از گرادیان، مشخصات حساسیت دریافت ناهمگن و آناتومی و محل قرارگیری بیمار هستند.

به عنوان مثال تصاویر زیر را در نظر بگیرید:



تصویر سمت چپ، تصویر اصلی ثبت شده را نمایش می‌دهد که به وضوح با مشکل ناهمگنی شدت میدان مواجه است. در تصویر وسط، نقشه تقریبی ناهمگنی شدت میدان مربوط به این تصویر مشاهده می‌شود. تصویر سمت راست، معرف تصویر تصحیح شده است که با استفاده از نقشه تقریبی ناهمگنی شدت میدان، بدست آمده است.

بر اساس نظریه نگاشت میدان فرکانس رادیویی (RF)، این ناهمگنی شدت میدان در تصاویر MR را می‌توان به صورت ضرب شونده مدل کرد. برای این موضوع، مدل‌های متنوعی در نظر گرفته شده است که به شرح زیر هستند:

$$\mathrm{v}(\mathrm{x}) = \mathrm{u}(\mathrm{x})\mathrm{b}(\mathrm{x}) + \mathrm{n}(\mathrm{x})$$

$$\mathrm{v}(\mathrm{x}) = \big(\mathrm{u}(\mathrm{x}) + \mathrm{n}(\mathrm{x})\big)\mathrm{b}(\mathrm{x})$$

$$\log\big(\mathrm{v}(\mathrm{x})\big) = \log\big(\mathrm{u}(\mathrm{x})\big) + \log\big(\mathrm{b}(\mathrm{x})\big) + \mathrm{n}'(\mathrm{x})$$

در عبارت‌های فوق، $\mathrm{v}(\mathrm{x})$ تصویر اصلی ثبت شده، $\mathrm{u}(\mathrm{x})$ تصویر واقعی، $\mathrm{b}(\mathrm{x})$ ناهمگنی شدت میدان، $\mathrm{n}(\mathrm{x})$ نویز جمع شونده و $\mathrm{n}'(\mathrm{x})$ لگاریتم نویز جمع شونده هستند.

در روش AFCM تابع هزینه به صورت زیر تعریف می‌شود:

$$J_{AFCM} = \sum_{i,j}\sum_{k=1}^{C} u_k^2(i,j)\|y(i,j) - m(i,j)v_k\|^2 + \cdots$$

$$\lambda_1 \sum_{i,j}\left((m(i,j) * D_i)^2 + \left(m(i,j) * D_j\right)^2\right) + \lambda_2 \sum_{i,j}\left((m(i,j) ** D_{ii})^2 + 2\left(m(i,j) * D_{ij}\right)^2 + \left(m(i,j) ** D_{jj}\right)^2\right)$$

با مقایسه تابع هزینه این روش با تابع هزینه روش FCM، دو تغییر اساسی مشهود است.

۱) متغیرهای $m(i,j)$ که برای مدل کردن ناهمگنی شدت میدان هستند.

۲) دو عبارت regularization که برای اعمال شرط همواری بر روی ناهمگنی شدت میدان اضافه شده اند.

در واقع در روش AFCM هدف این است که دو عمل تعیین ناهمگنی شدت میدان و خوشه‌بندی به صورت هم‌زمان انجام شوند. بدین منظور، متغیرهای $m(i,j)$ برای حذف تاثیر ناهمگنی شدت میدان اضافه شده اند؛ اما اعمال این متغیرهای جدید بدون در نظر گرفتن روشی برای کنترل همواری ناهمگنی شدت میدان، باعث می‌شود مسئله جوابی بدیهی داشته باشد. لذا دو عبارت regularization نیز مورد استفاده قرار گرفته اند تا با در نظر گرفتن مقدار مشتق‌های اول و دوم متغیرهای $m(i,j)$ از ناهموار شدن ناهمگنی شدت میدان جلوگیری کنند.

مجهولات مورد نظر را می‌توان با استفاده از یک الگوریتم Iterative بدست آورد. مراحل الگوریتم به شرح زیر است:

1. initial guess for $\{v_k\}_{k=1}^{C}$ using K-means, FCM, … and set $m(i,j) = 1$

2. Compute membership functions: $u_k(i,j) = \frac{\|y(i,j)-m(i,j)v_k\|^{-2}}{\sum_{l=1}^{C}\|y(i,j)-m(i,j)v_l\|^{-2}}$

3. Compute centroids: $v_k = \frac{\sum_{i,j} u_k^2(i,j)y(i,j)m(i,j)}{\sum_{i,j} u_k^2(i,j)m(i,j)}$

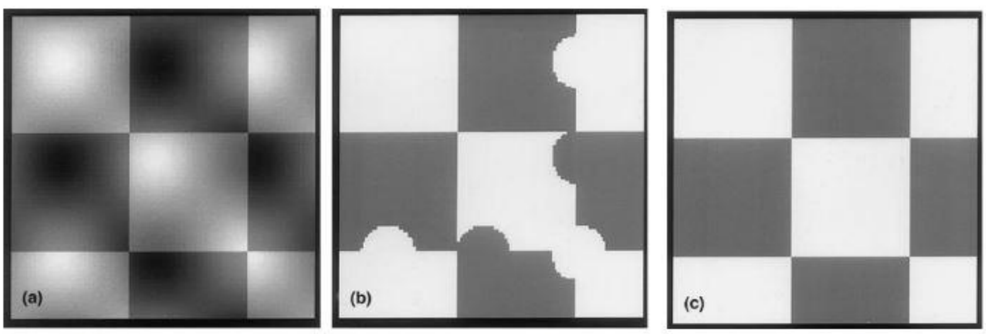
4. Compute multiplier field:

$$y(i,j)\sum_{k=1}^{C} u_k^2(i,j)v_k = m(i,j)\sum_{k=1}^{C} u_k^2(i,j)v_k^2 + \lambda_1\left(m(i,j) ** H_1(i,j)\right) + \lambda_2\left(m(i,j) ** H_2(i,j)\right)$$

$$\text{where} \quad H_1(i,j) = D_i * \breve{D}_i + D_j * \breve{D}_j, \quad H_1(i,j) = D_{ii} * \breve{D}_{ii} + 2\left(D_{ij} * \breve{D}_{ij}\right) + D_{jj} * \breve{D}_{jj}$$

5. step over 2-3-4 until convergence

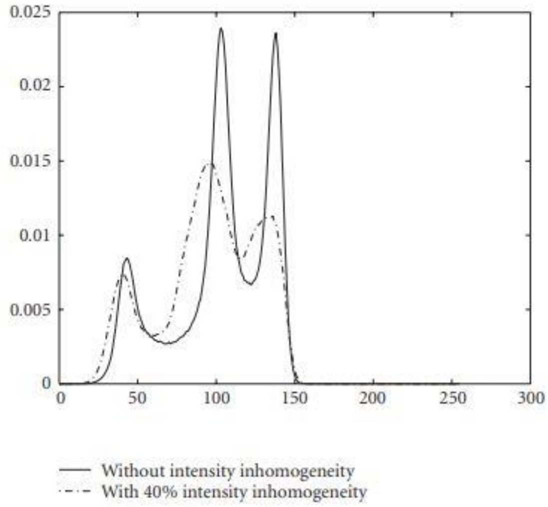
نمونه‌ای از خروجی این روش در مقایسه با روش FCM، به صورت زیر است:



(a) original (b) FCM (c) AFCM

سوال ۲: در مقاله مذکور، روش‌های متعددی جهت بازسازی نقشه مربوط به ناهمگنی شدت میدان و تصحیح تصویر ثبت شده، ارائه شده است. در ادامه دو روش Entropy minimization و Bayesian framework بررسی می‌شوند.

روش Entropy minimization: آنتروپی، به عنوان یک معیار مکررا استفاده شده برای توصیف توزیع شدت روشنایی تصویر، در طراحی الگوریتم‌های ترمیم تصویر، آستانه‌گذاری بر روی تصویر و طبقه‌بندی تصویر مورد استفاده قرار گرفته است. این معیار برای کمی کردن خصوصیات تصویر در حضور ناهمگنی شدت میدان و هدایت نمودن جستجوی پارامترهای موجود جهت حذف ناهمگنی شدت میدان نیز کاربرد دارد. در واقع فرض می‌شود که توزیع شدت روشنایی تصویر ثبت شده اصلی، multimodal بوده است و حضور ناهمگنی شدت میدان، باعث ایجاد overlap بین شدت روشنایی بخش‌های مختلف می‌شود. برای درک بهتر این موضوع هیستوگرام تصویری از BrainWeb-simulated را در نظر بگیرید:



در تصویر فوق، خط پر مربوط به هیستوگرام شدت روشنایی تصویر واقعی و خط نقطه‌چین مربوط به هیستوگرام شدت روشنایی تصویر در حضور ناهمگنی شدت میدان است. همان‌طور که مشخص است، حضور ناهمگنی شدت میدان، دره‌های میان مدهای مختلف را به شدت صاف کرده است. افزایش میزان صافی هیستوگرام، باعث افزایش آنتروپی می‌شود. در نتیجه برای حذف ناهمگنی شدت میدان، می‌توان پارامترهای موجود را به گونه‌ای جستجو کرد که آنتروپی تصویر کاهش یابد.

روش Bayesian framework: در مواردی که برای ناهمگنی شدت میدان یک توزیع احتمال در نظر گرفته شده است، قانون بیز مکررا برای تخمین ناهمگنی شدت میدان مورد استفاده قرار گرفته است. فرض کنید β یک بردار تصادفی به صورت $(\beta_1, \dots, \beta_n)$ با توزیع احتمال $p(\beta)$ باشد. هدف تخمین β از روی بیشینه کردن احتمال شرطی β در صورت داشتن y است. یعنی داریم:

$$\hat{\beta} = \max_{\beta} p(\beta|y)$$

بر اساس قانون بیز عبارت داریم:

$$\hat{\beta} = \max_{\beta} p(y|\beta)p(\beta)$$

با در نظر گرفتن توزیع احتمال گوسی برای $p(y|\beta)$ و $p(\beta)$ در پیکسل‌های مختلف داریم:

$$p(\beta) = G_{\psi_\beta}(\beta)$$

$$p(y_i|\Gamma_i, \beta_i) = G_{\psi_{\Gamma_i}}(y_i - \mu(\Gamma_i) - \beta_i)$$

که در عبارت‌های فوق داریم:

$$G_{\psi_x}(x) = (2\pi)^{-\frac{n}{2}} |\psi_x|^{-\frac{1}{2}} \exp\left(-\frac{1}{2} x^T \psi_x^{-1} x\right)$$

همچنین Γ_i کلاس پیکسل i و $\mu(\Gamma_i)$ میانگین این کلاس است.

با در نظر گرفتن فرض استقلال برای شدت روشنایی پیکسل‌های مختلف داریم:

$$p(y|\beta) = \prod_i p(y_i|\beta_i) = \prod_i \sum_{\Gamma_i} p(y_i|\Gamma_i, \beta_i) p(\Gamma_i)$$

در تصویرهای واقعی بسیار محتمل است که هیستوگرام تصویر، از توزیع ترکیب گوسی (GMM) پیروی نکند. یک ایده برای اصلاح روش مذکور، بر مبنای معرفی کلاس Γ_{other} که دارای توزیع غیرگوسی است، به صورت زیر است:

$$p(y_i|\beta_i) = \sum_{\Gamma_i} p(y_i|\Gamma_i, \beta_i) p(\Gamma_i) + \lambda p(\Gamma_{other})$$

با این اصلاح، تخمین ناهمگنی شدت میدان، تنها بر اساس کلاس‌های گوسی صورت می‌گیرد.

بخش شبیه‌سازی

سوال ۱: الف) در روش FCM استاندارد تابع هزینه برای تقسیم $\{x_k\}_{k=1}^{N}$ به C خوشه به صورت زیر تعریف می‌شود:

$$J_{FCM} = \sum_{i=1}^{C}\sum_{k=1}^{N} u_{ik}^{p}\|x_k - v_i\|^2$$

که در آن $\{u_{ik}\}_{i=1,k=1}^{C,N}$، متغیرهای membership معرف میزان تعلق نمونه k به خوشه i و $\{v_i\}_{i=1}^{C}$، متغیرهای prototype معرف مرکز خوشه‌های مذکور هستند. برای متغیرهای membership شروط زیر باید برقرار باشند:

$$u_{ik} \in [0,1], \sum_{i=1}^{C} u_{ik} = 1\ \forall k, 0 < \sum_{k=1}^{N} u_{ik} < N\ \forall i$$

در روش BCFCM، با معرفی عبارتی برای ایجاد امکان تاثیرپذیری برچسب یک پیکسل از برچسب‌های پیکسل‌های همسایه، تابع هزینه فوق، به صورت زیر اصلاح می‌شود:

$$J_{BCFCM} = \sum_{i=1}^{C}\sum_{k=1}^{N} u_{ik}^{p}\|x_k - v_i\|^2 + \frac{\alpha}{N_R}\sum_{i=1}^{C}\sum_{k=1}^{N} u_{ik}^{p}\left(\sum_{r\in\mathcal{N}_k}\|x_r - v_i\|^2\right)$$

که در آن $\mathcal{N}_k$ مجموعه معرف همسایه‌های x_k و N_R کاردینالیته این مجموعه است. میزان تاثیر پیکسل‌های همسایه توسط پارامتر α کنترل می‌شود. هر چه میزان SNR تصویر کمتر باشد، باید مقادیر بزرگ‌تری برای پارامتر α انتخاب کرد.

حال با در نظر گرفتن یک تبدیل لگاریتمی روی شدت روشنایی پیکسل‌های تصویر، می‌توان ناهمگنی شدت میدان را به صورت یک عبارت جمع شونده به صورت زیر مدل کرد:

$$y_k = x_k + \beta_k \quad \forall k \in \{1, \dots, N\}$$

که در آن y_k لگاریتم شدت روشنایی در پیکسل k ام تصویر ثبت شده، x_k لگاریتم شدت روشنایی در پیکسل k ام تصویر واقعی و β_k لگاریتم اثر ناهمگنی شدت میدان در پیکسل k ام است.

با جایگزین کردن x_k با استفاده از رابطه فوق، تابع هزینه روش BCFCM به صورت زیر بدست می‌آید:

$$J_{BCFCM} = \sum_{i=1}^{C}\sum_{k=1}^{N} u_{ik}^{p}\|y_k - \beta_k - v_i\|^2 + \frac{\alpha}{N_R}\sum_{i=1}^{C}\sum_{k=1}^{N} u_{ik}^{p}\left(\sum_{r\in\mathcal{N}_k}\|y_r - \beta_r - v_i\|^2\right)$$

در نتیجه تابع هزینه مسئله بهینه‌سازی مقید به صورت زیر است:

$$J_{BCFCM} = \sum_{i=1}^{C}\sum_{k=1}^{N} u_{ik}^{p}\|y_k - \beta_k - v_i\|^2 + \frac{\alpha}{N_R}\sum_{i=1}^{C}\sum_{k=1}^{N} u_{ik}^{p}\left(\sum_{r\in\mathcal{N}_k}\|y_r - \beta_r - v_i\|^2\right) + \lambda\left(1 - \sum_{i=1}^{C} u_{ik}\right)$$

پارامترهای membership، همان متغیرهای $\{u_{ik}\}_{i=1,k=1}^{C,N}$ هستند که با استفاده از مشتق‌گیری از تابع هزینه به صورت زیر بدست می‌آیند:

$$u_{ik}^{*} = \left(\frac{\lambda}{p\left(\|y_k - \beta_k - v_i\|^2 + \frac{\alpha}{N_R}\sum_{r\in\mathcal{N}_k}\|y_r - \beta_r - v_i\|^2\right)}\right)^{\frac{1}{p-1}}$$

$$\sum_{i=1}^{C}\left(\frac{\lambda}{p\left(\|y_k-\beta_k-v_i\|^2+\frac{\alpha}{N_R}\sum_{r\in\mathcal{N}_k}\|y_r-\beta_r-v_i\|^2\right)}\right)^{\frac{1}{p-1}}=1$$

$$\lambda=\frac{p}{\left(\sum_{i=1}^{C}\left(\frac{1}{\|y_k-\beta_k-v_i\|^2+\frac{\alpha}{N_R}\sum_{r\in\mathcal{N}_k}\|y_r-\beta_r-v_i\|^2}\right)^{\frac{1}{p-1}}\right)^{p-1}}$$

$$u_{ik}^*=\frac{1}{\sum_{j=1}^{C}\left(\frac{\|y_k-\beta_k-v_i\|^2+\frac{\alpha}{N_R}\sum_{r\in\mathcal{N}_k}\|y_r-\beta_r-v_i\|^2}{\left\|y_k-\beta_k-v_j\right\|^2+\frac{\alpha}{N_R}\sum_{r\in\mathcal{N}_k}\left\|y_r-\beta_r-v_j\right\|^2}\right)^{\frac{1}{p-1}}}$$

پارامترهای cluster prototype، همان متغیرهای $\{\nu_i\}_{i=1}^{C}$ هستند که با استفاده از مشتق‌گیری از تابع هزینه به صورت زیر بدست می‌آیند:

$$\nu_i^*=\frac{\sum_{k=1}^{N}u_{ik}^p\left((y_k-\beta_k)+\frac{\alpha}{N_R}\sum_{r\in\mathcal{N}_k}(y_r-\beta_r)\right)}{(1+\alpha)\sum_{k=1}^{N}u_{ik}^p}$$

متغیرهای bias field، همان متغیرهای $\{\beta_k\}_{k=1}^{N}$ هستند که با استفاده از مشتق‌گیری از تابع هزینه به صورت زیر بدست می‌آیند:

$$\beta_k^*=y_k-\frac{\sum_{i=1}^{C}u_{ik}^p\nu_i}{\sum_{i=1}^{C}u_{ik}^p}$$

مجهولات مورد نظر را می‌توان با استفاده از یک الگوریتم Iterative بدست آورد. مراحل الگوریتم به شرح زیر است:

1. Select initial cluster prototypes $\{\nu_i\}_{i=1}^{C}$. Set $\{\beta_k\}_{k=1}^{N}$ to equal and very small values, for example 0.01.

2. Update membership parameters $\{u_{ik}\}_{i=1,k=1}^{C,N}$.

$$u_{ik}^*=\frac{1}{\sum_{j=1}^{C}\left(\frac{\|y_k-\beta_k-v_i\|^2+\frac{\alpha}{N_R}\sum_{r\in\mathcal{N}_k}\|y_r-\beta_r-v_i\|^2}{\left\|y_k-\beta_k-v_j\right\|^2+\frac{\alpha}{N_R}\sum_{r\in\mathcal{N}_k}\left\|y_r-\beta_r-v_j\right\|^2}\right)^{\frac{1}{p-1}}}$$

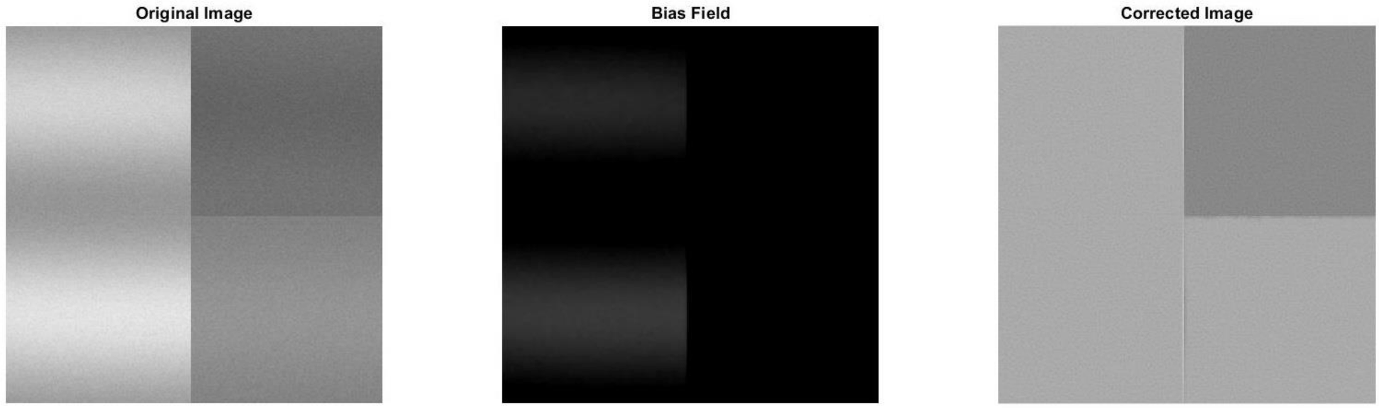
3. Update cluster prototype parameters $\{\nu_i\}_{i=1}^{C}$.

$$\nu_i^*=\frac{\sum_{k=1}^{N}u_{ik}^p\left((y_k-\beta_k)+\frac{\alpha}{N_R}\sum_{r\in\mathcal{N}_k}(y_r-\beta_r)\right)}{(1+\alpha)\sum_{k=1}^{N}u_{ik}^p}$$

4. Update bias field parameters $\{\beta_k\}_{k=1}^{N}$.

$$\beta_k^*=y_k-\frac{\sum_{i=1}^{C}u_{ik}^p\nu_i}{\sum_{i=1}^{C}u_{ik}^p}$$

ب) با استفاده از تابع BCFCM2D و در نظر گرفتن دو خوشه، موارد خواسته شده را انجام می‌دهیم. خروجی به صورت زیر بدست می‌آید:



سوال ۲: الف) به طور سنتی، یک snake، یک خم $X(s) = [x(s), y(s)],\ \ s \in [0, 1]$ است که در مختصات مکانی تصویر به گونه‌ای حرکت می‌کند که تابع انرژی زیر کمینه شود:

$$E = \int_0^1 \frac{1}{2}(\alpha|X'(s)|^2 + \beta|X''(s)|^2) + E_{ext}\big(X(s)\big)ds$$

در عبارت فوق α و β، پارامترهای وزن‌دهی برای کنترل میزان tension و rigidity در snake مورد نظر هستند. همچنین تابع انرژی خارجی E_{ext} از روی تصویر به گونه‌ای بدست می‌آید که بر روی نواحی مورد علاقه، همچون مرزهای تصویر، مقادیر کوچک‌تر خود را کسب کند.

با در نظر گرفتن یک تصویر gray-level همچون I(x, y)، تابع‌های انرژی خارجی معمول، که برای هدایت contour فعال به سمت لبه‌های پله مانند طراحی شده اند، عبارت اند از:

$$E_{ext}^1(x, y) = -|\nabla I(x, y)|^2$$

$$E_{ext}^2(x, y) = -\left|\nabla\big(G_\sigma(x, y) * I(x, y)\big)\right|^2$$

اگر تصویر یک line drawing (ترسیم خط) سیاه بر روی صفحه سفید باشد، تابع‌های انرژی خارجی مناسب، شامل موارد زیر خواهد بود:

$$E_{ext}^3(x, y) = I(x, y)$$

$$E_{ext}^4(x, y) = G_\sigma(x, y) * I(x, y)$$

اگر یک snake، تابع انرژی E را کمینه کند، در معادله Euler نیز صدق می‌کند:

$$\underbrace{\alpha X'(s) + \beta X''(s)}_{F_{int}} \underbrace{-\nabla E_{ext}}_{F_{ext}^{1}} = 0$$

در عبارت فوق نیروی داخلی F_{int} با کشیدگی و خمیدگی مخالفت می‌کند، در حالی که نیروی خارجی F_{ext}^{1}، snake را به سمت contour تصویر مطلوب می‌کشد.

برای حل این معادله، با در نظر گرفتن X به صورت تابعی از t، علاوه بر s، snake مورد نظر پویا می‌شود و سپس مشتق جزئی X نسبت به t، برابر با سمت چپ معادله قرار می‌گیرد:

$$X_t(s,t) = \alpha X'(s,t) + \beta X''(s,t) - \nabla E_{ext}$$

وقتی $X(s,t)$ پایدار شود، عبارت $X_t(s,t)$ ناپدید می‌شود و جواب معادله بدست می‌آید.

ب) میدان gradient vector flow (GVF) به صورت یک میدان برداری $w(x,y) = \big(u(x,y), v(x,y)\big)$ تعریف می‌شود که تابع انرژی زیر را کمینه می‌کند:

$$\varepsilon = \iint \mu\big(u_x^2 + u_y^2 + v_x^2 + v_y^2\big) + |\nabla f|^2 |w - \nabla f|^2 dxdy$$

هدف فرمول بالا، هموار کردن نتیجه تنها در مکان‌هایی است که داده‌ای وجود ندارد. به طور خاص، وقتی $|\nabla f|$ کوچک است، عبارت غالب در تابع انرژی، مشتق‌های جزئی هستند که منجر به یک میدان هموار می‌شود. اما وقتی $|\nabla f|$ بزرگ است، عبارت $|\nabla f|^2 |w - \nabla f|^2$، عبارت غالب در تابع انرژی خواهد بود که با قرار دادن $w = \nabla f$ کمینه می‌گردد. پارامتر μ یک پارامتر regularization برای کنترل tradeoff میان عبارت‌های اول و دوم است. هر چه مقدار نویز بیشتر باشد، باید مقدار μ نیز بزرگ‌تر انتخاب شود.

با استفاده از calculus of variations، می‌توان نشان داد که میدان GVF کمینه کننده تابع انرژی فوق، با حل معادلات Euler که در زیر آمده اند، بدست می‌آید:

$$\mu\nabla^2 u - (u - f_x)\big(f_x^2 + f_y^2\big) = 0$$
$$\mu\nabla^2 v - \big(v - f_y\big)\big(f_x^2 + f_y^2\big) = 0$$

در عبارت فوق ∇^2، عملگر لاپلاسین است.

در نواحی همگن، عبارت دوم هر دو معادله صفر است و در نتیجه در این نواحی u و v بر اساس معادله لاپلاسین تعیین می‌شوند.

جواب معادله‌های فوق با در نظر گرفتن u و v به صورت تابعی از t، علاوه بر x و y، و حل معادلات جایگزین زیر بدست می‌آید:

$$u_t(x,y,t) = \mu\nabla^2 u(x,y,t) - \big(u(x,y,t) - f_x(x,y)\big)\left(f_x^2(x,y) + f_y^2(x,y)\right)$$
$$v_t(x,y,t) = \mu\nabla^2 v(x,y,t) - \left(v(x,y,t) - f_y(x,y)\right)\left(f_x^2(x,y) + f_y^2(x,y)\right)$$

وقتی $u(x,y,t)$ و $v(x,y,t)$ پایدار شوند، عبارت‌های $u_t(x,y,t)$ و $v_t(x,y,t)$ ناپدید می‌شوند و جواب معادلات بدست می‌آید. همچنین بر اساس رابطه $w(x,y) = \big(u(x,y), v(x,y)\big)$، میدان GVF تعیین می‌گردد.

ج) تابع Snake2D دارای سه ورودی و دو خروجی است. ورودی اول تصویر مورد نظر، ورودی دوم یک لیست N*2 شامل نقاط contour اولیه و ورودی سوم یک struct شامل تنظیمات پارامترهای مختلف است. خروجی اول یک لیست M*2 شامل نقاط contour نهایی و خروجی دوم تصویر باینری بخش‌بندی شده است. این تابع در ابتدا با استفاده از دو تابع MakeContourClockWise و InterpolateContourPoints2D و پارامتر nPoints، از روی contour اولیه، contour دیگری در جهت عقربه‌های ساعت می‌سازد که دارای nPoints نقطه است. در ادامه با استفاده از تابع ExternalForceImage2D، تصویر مورد نظر و پارامترهای Wline، Wedge، Wterm و sigma1، تابع انرژی E_{ext} را محاسبه می‌کند. سپس با استفاده از تابع ImageDerivatives2D و پارامتر sigma2، گرادیان تابع انرژی E_{ext} که توسط یک فیلتر گوسی هموار شده است را بدست می‌آورد. حال با استفاده از تابع GVFOptimizeImageForce2D، پارامترهای Mu، Giterations و sigma3 و خروجی قسمت قبل، میدان GVF محاسبه می‌گردد. در انتها با استفاده از دو تابع SnakeInternalForceMatrix2D و SnakeMoveIteration2D، پارامترهای Iterations، Gamma، Kappa و Delta و میدان GVF حاصل از قسمت قبل، snake مورد نظر در چندین مرحله به‌روزرسانی می‌شود. خروجی تابع با استفاده از تابع DrawSegmentedArea2D و snake نهایی بدست می‌آید. شایان ذکر است در صورتی که پارامتر verbose برابر یک باشد، contour اولیه بر روی تصویر مورد نظر، تابع انرژی E_{ext}، میدان GVF و تغییرات snake در مراحل مختلف رسم می‌شوند.

پارامترهای ورودی به صورت زیر هستند:

```
options (general),
Options.Verbose: If true show important images, default false
Options.nPoints: Number of contour points, default 100
Options.Gamma: Time step, default 1
Options.Iterations: Number of iterations, default 100

options (Image Edge Energy / Image force))
Options.Sigma1: Sigma used to calculate image derivatives, default 10
Options.Wline: Attraction to lines, if negative to black lines otherwise white
lines, default 0.04
Options.Wedge: Attraction to edges, default 2
Options.Wterm: Attraction to terminations of lines (end points) and corners,
default 0.01
Options.Sigma2: Sigma used to calculate the gradient of the edge energy image
(which gives the image force), default 20

options (Gradient Vector Flow)
Options.Mu: Tradeoff between real edge vectors, and noise vectors,
default 0.2. (Setting this to high, > 0.5, gives an unstable Vector Flow)
Options.GIterations: Number of GVF iterations, default 0
Options.Sigma3: Sigma used to calculate the laplacian in GVF, default 1

options (Snake)
Options.Alpha: Weight of Membrame energy  (first order), default 0.2
Options.Beta: Weight of Thin plate energy (second order), default 0.2
Options.Delta: Weight of Baloon force, default 0.1
Options.Kappa: Weight of external image force, default 2
```

پارامترهای GVF:

Mu: پارامتر regularization برای کنترل tradeoff میان بردارهای لبه واقعی و بردارهای نویز در محاسبه GVF

Giterations: تعداد تکرار در محاسبه GVF

Sigma3: انحراف معیار استفاده شده برای بدست آوردن لاپلاسین در محاسبه GVF

پارامترهای Snake:

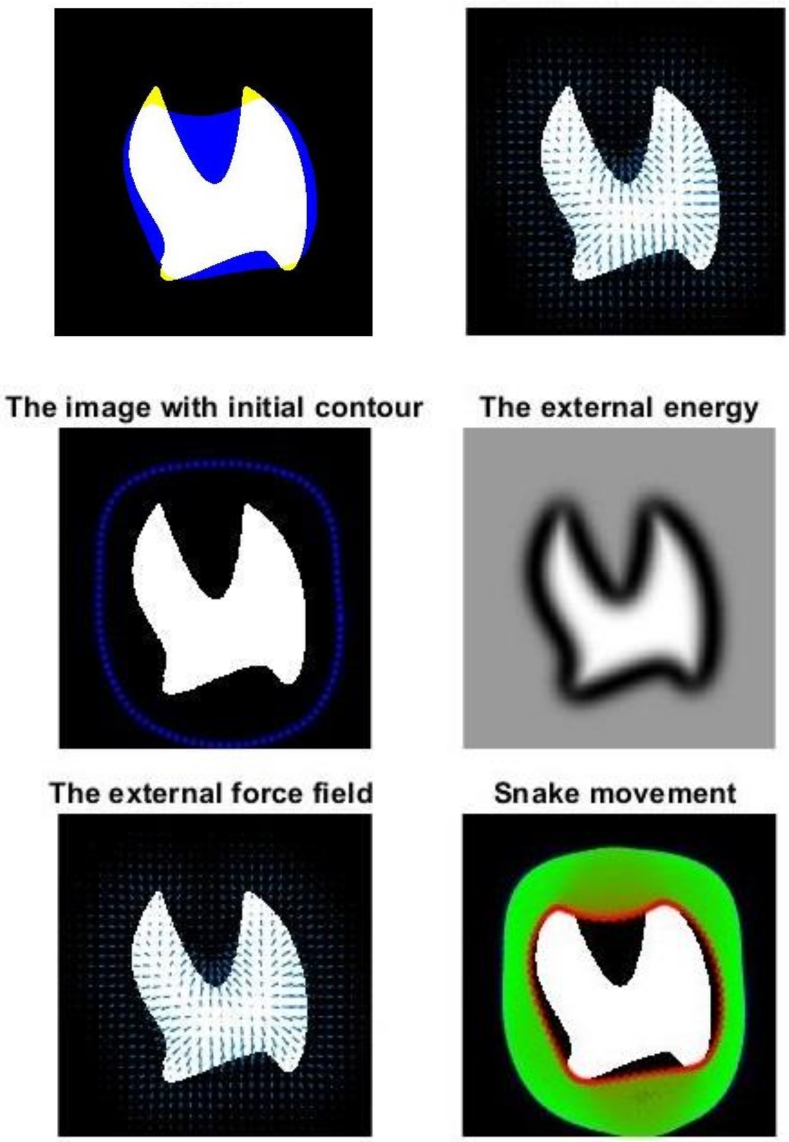
Alpha: ضریب انرژی Membrame در به‌روزرسانی snake

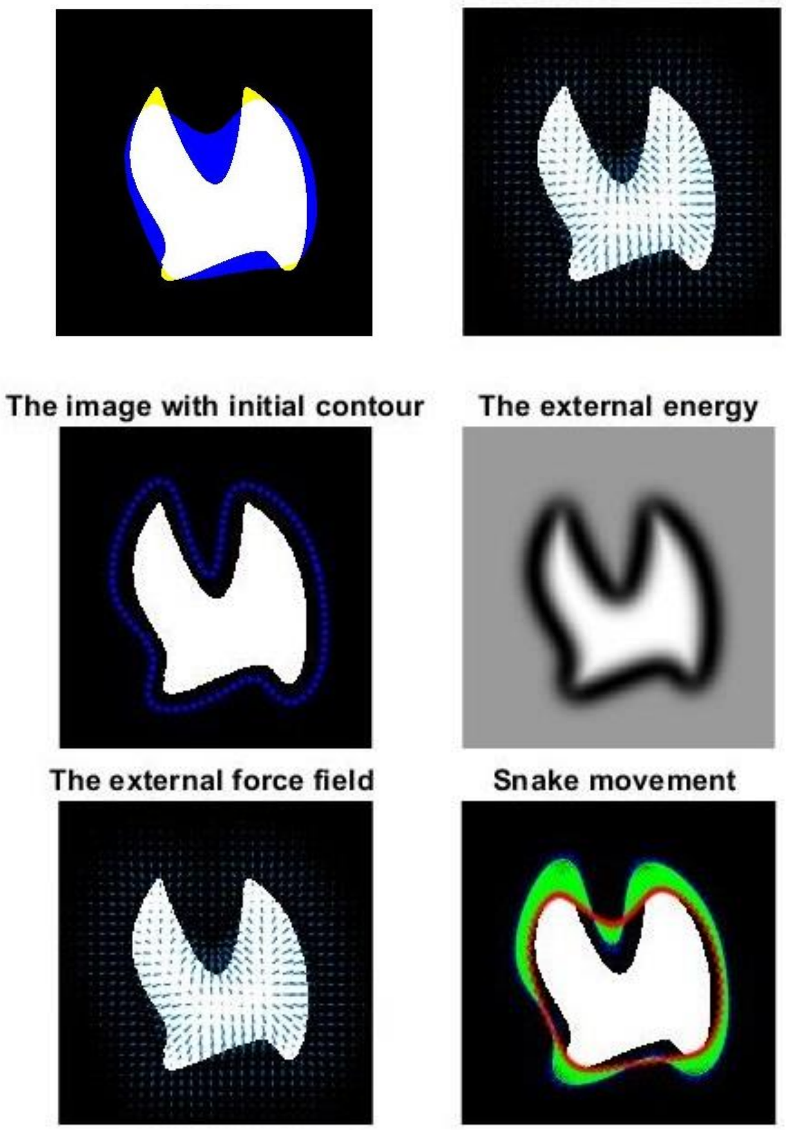
Beta: ضریب انرژی Thin plate در به‌روزرسانی snake

Delta: ضریب نیروی Baloon در به‌روزرسانی snake

Kappa: ضریب نیرو خارجی تصویر در به‌روزرسانی snake

د) با استفاده از تابع Snake2D و انتخاب پارامترهای مناسب، موارد خواسته شده را انجام می‌دهیم. به ازای دو contour اولیه متفاوت، خروجی به صورت زیر بدست می‌آید:





همان‌طور که مشاهده می‌شود خروجی‌های حاصل به ازای دو contour اولیه متفاوت، تقریبا یکسان هستند. در مجموع با توجه به ساختار شکل و تورفتگی‌های آن، اگر snake مورد نظر بخواهد وارد یک تورفتگی شود، بیرون‌زدگی‌های اطراف آن تورفتگی، از بین می‌روند. اما از آنجایی که خروجی GVF یک میدان برداری است، می‌تواند تورفتگی‌های شکل را به نحو بهتری نمایش دهد.

در انتها شایان ذکر است که برحسب نحوه انتخاب contour اولیه و تنظیم پارامترهای تابع Snake2D، ممکن است جواب‌های متفاوتی برای هر دو روش حاصل شود.