BAB II

TINJAUAN PUSTAKA

Pada bab ini, peneliti akan membahas mengenai penelitian terkait dan teori yang akan digunakan dalam segmentasi citra medis menggunakan *fuzzy level set* berbasis komputasi parallel GPU CUDA. Berbagai penelitian telah dilakukan untuk melakukan segmentasi citra medis.

A. Penelitian Terdahulu

Penelitian terkait *fuzzy level set* telah dilakukan oleh berbagai peneliti dalam proses kluster nilai parameter awal level set dengan metode *spatial fuzzy clustering* (Li, et al., 2011), (Ghalehnovi & Zahedi, 2014), (Anami & Unki, 2013), (Anitha & Peter, 2015). Rangkuman penelitian terkait dapat dilihat pada tabel 2.1.

Tabel 2.1 Rangkuman Penelitian Terdahulu

Aspek	***(Li, et al., 2011)***	***(Ghalehnovi & Zahedi, 2014)***	***(Anami & Unki, 2013)***	***(Anitha & Peter, 2015)***	***Airlangga, 2016 ****
Domain/objek	*Ultrasound, MRI, CT*	*Angiogram*	*MRI*	*Mammogram*	*MRI, CT*
Data	*Otak, Liver, Jaringan kanker*	*Bilik jantung*	*Otak*	*Jaringan Syaraf*	*Otak, Liver*
Metode	*Spatial Fuzzy Level Set*	*Fuzzy Level Set*	*Fuzzy Level Set*	*Spatial Fuzzy Level Set*	*Fuzzy Distance Regularization Level Set, Fuzzy Chanvese Level Set*
Tool	*Matlab*	*Matlab*	*Matlab*	*Matlab*	*NVIDIA CUDA C*
Hasil	*Konvergen pada iterasi ke 300*	*Konvergen pada iterasi ke 500*	*Konvergenit erasi ke 100*	*Konvergen pada iterasi ke 500*	**sedang dilakukan*

B. Landasan Teori

Penelitian yang diusulkan menggunakan pendekatan *Partial Differential Equation* (PDE) dengan memanfaatkan komputasi parallel GPU CUDA. Berikut ini adalah landasan teori-teori yang digunakan oleh peneliti di dalam penelitian ini.

1. Citra Medis

Citra merupakan gambar yang merepresentasikan sesuatu. Citra dapat berupa gambar dari seseorang, orang banyak atau hewan, atau suatu pemandangan luar, atau *microphotograph* dari suatu komponen elektronik atau juga hasil dari pencitraan medis (Kadir & Susanto, 2013). Pencitraan medis merupakan citra yang dibuat dengan menggunakan beberapa teknologi yang berbeda yang digunakan untuk melihat tubuh manusia dan bertujuan untuk mendiagnosa, memonitor dan atau memeriksa kondisi medis. Setiap teknologi memberikan informasi yang berbeda mengenai area dari tubuh manusia yang dipelajari atau diperiksa, yang berhubungan dengan penyakit, kecelakaan atau untuk melihat perkembangan pemeriksaan medis.

Salah satu alat yang digunakan untuk membuat citra medis adalah dengan menggunakan MRI. MRI menciptakan medan magnet yang temporer di sekitar tubuh pasien. Gelombang radio dikirimkan dan diterima dengan *transmitter/receiver* dalam mesin kemudian sinyal tersebut membuat citra digital dari area yang diinginkan.

2. Segmentasi Citra

Segmentasi merupakan proses untuk membagi citra digital menjadi beberapa segmen (Kadir & Susanto, 2013). Tujuan untuk melakukan segmentasi citra adalah dengan membuat citra tersebut agar dapat lebih berarti dan lebih mudah untuk dianalisa (Cremers, et al., 2007). Terdapat banyak metode untuk melakukan segmentasi citra, yaitu *Intensity Thresholding* (Banerjee, et al., 2015), *Region Growing and Region Splitting* (LaTorre, et al., 2013), *Edge Detection* (Padmapriya, et al., 2012), *Watershed Segmentation* (Huang, et al., 2016), *Markov Random Model* (Zhang, et al., 2014), metode *Generalized Fuzzy C-means Clustering Algorithm* (Hardiyanto & Soelaiman, 2012), (Aparajeeta, et al., 2016), *Particle Swam Optimization* (Mandal, et al., 2014), *adaptive K-Means Clustering Algorithm* (Li, et al., 2015), *Distance Regularization Level Set Evolution* (DRLSE) (Li., et al., 2010).

3. Level Set

Metode level set memiliki berbagai variasi dalam melakukan segmentasi citra (Sethian, 1999), (Osher & Fedkiw, 2003). Segmentasi citra dengan mencari rata-rata dari *acitve contour* merupakan pendekatan yang dikenal dengan luas (Chan & Vese, 2001), (McInerney & Terzopoulos, 1996), (Wu., et al., 2009), namun selain karakteristik parametrik dari *active contour*, metode level set dipengaruhi oleh fungsi waktu. Hal ini memungkinkan untuk aproksimasi evolusi dari *active contour* yang secara implisit berkembang dari nilai level set awal yang disebut *zero level set*.

$$\phi(t, x, y) < 0\ (x, y) \text{ didalam } T(t) \quad (1)$$

$$\phi(t, x, y) = 0\ (x, y) \text{ pada } T(t) \quad (2)$$

$$\phi(t, x, y) > 0\ (x, y) \text{ diluar } T(t) \quad (3)$$

Keterangan:

ϕ : Phi Level Set (Daerah Segmentasi)

t : Waktu

x : Koordinat x

y : Koordinat y

T : Tepi Citra Asli

Dimana T terdiri dari *isocontour* rangkaian atau tunggal. Dalam segmentasi citra, hubungan ini dapat dituliskan sebagai berikut.

$$\cup Sk \cup T = I \quad (4)$$

Keterangan:

Sk : Citra Tersegmentasi

T : Tepi Citra

I : Citra Asli

Perlu diketahui bahwa keterlibatan dari variabel waktu menjadikan fungsi level set berubah ke dimensi yang lebih tinggi, meskipun metode ini memiliki keuntungan, waktu komputasi yang dihabiskan juga semakin bertambah. Sebagai contoh antarmuka *T* dapat dengan mudah ditentukan dengan mengecek nilai dari fungsi level set yang akan mengakomodasi

perubahan topologi dari antarmuka *T* secara alami. Dalam keadaan tertentu, evolusi nilai level set dipengaruhi oleh persamaan 5 dan 6.

$$\frac{\partial \phi}{\partial t} + F|\nabla \phi| = 0 \tag{5}$$

$$\phi(0, x, y) = \phi_0(x, y) \tag{6}$$

Keterangan:

$\frac{\partial \phi}{\partial t}$: Turunan Phi terhadap t

$F|\nabla \phi|$: Force (Gaya)

$\phi_0(x, y)$: Kontur Mula-Mula

$\phi(0, x, y)$: Kontur Sumbu x dan y

Dimana nilai $|\nabla \phi|$ adalah arah, $\phi_0(x, y)$ merupakan *initial contour* dan *F* merepresentasikan tekanan, termasuk tekanan internal geometri (misalnya: *mean curvature, contour length* dan area) dan tekanan eksternal dari gradien citra dan atau *artificial momentum* (Li, et al., 2006). Nilai *F* harus diregularisasi oleh fungsi tepi *g* untuk menghentikan iterasi dari level set. Persamaan *g* tersebut dapat dirumuskan seperti persaamaan 7.

$$g = \frac{1}{1 + |\nabla(G_\sigma * I)|^2} \tag{7}$$

Keterangan:

g : Penapisan Derau Gaussian

$\nabla(G_\sigma * I)$: Konvolusi Citra Asli dengan Gaussian

Dimana $G_\sigma * I$ merupakan konvolusi dari citra I dengan kernel gaussian. Fungsi g akan mendekati nilai nol di tepi/batasan, namun selain itu nilainya akan positif. Rumus berikut merupakan persamaan yang paling banyak digunakan dalam segmentasi citra menggunakan level set (Caseless, et al., 1997).

$$\frac{\partial \phi}{\partial t} = g|\nabla\phi|(\text{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right) + v \quad (8)$$

Keterangan:

$\frac{\partial \phi}{\partial t}$: Turunan phi terhadap t

$g|\nabla\phi|$: Konvolusi Gaussian dengan Phi

$(\text{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right)$: Kelengkungan Phi

v : Kecepatan Perambatan Kontur Segmentasi

Dimana $\text{div}(\nabla\phi/\nabla\phi)$ merupakan fungsi aproksimasi lengkungan rata-rata k. Sedangkan v adalah kecepatan pengembangan nilai level set. Dalam metode level set, kontur (atau lebih umum dikenal sebagai *hypersurface*) dari

suatu daerah, pada fungsi zero level set. Meskipun hasil akhir dari metode level set adalah zero level set pada fungsi level set, sangat penting untuk mempertahankan fungsi level set pada sebuah kondisi yang baik, artinya evolusi level set berjalan dengan stabil dan memiliki akurasi komputasi numerik yang akurat.

Hal ini berarti menunjukkan bahwa fungsi level set berjalan dengan lancar, tidak terlalu terjal atau terlalu datar (paling minimum bergerak di sekitar zero level set) selama evolusi level set. Kondisi ini cukup memuaskan dengan memberikan *signed distance function* untuk nilai unik dimana $|\nabla\phi| = 1$ yang berarti ditunjukkan sebagai sifat *signed distance*. Untuk kasus objek 2 dimensi (citra), kita mempertimbangkan *signed distance function* $z = \phi(x, y)$ sebagai sebuah permukaan. Selain itu, bidang singgung dibuat setara dengan sudut 45 derajat yang mana bidang xy dan z, dapat dengan mudah diverifikasi oleh sifat *signed distance* $|\nabla\phi| = 1$.

Untuk sifat yang diinginkan ini, fungsi *signed distance* telah secara luas digunakan sebagai fungsi level set di dalam metode level set. di dalam rumus level set konvensional, fungsi level set secara khas diinisialisasikan dan direinisialisasi secara periodik sebagai *signed distance function* (Li., et al., 2010).

Lebih dari itu, terdapat masalah baik secara teori dan praktek di dalam formula level set konvensional mengenai reinisialisasi dalam level set tradisional. Di dalam persamaan evolusi level set yang dituliskan pada bentuk

persamaan 5 dimana fungsi kecepatan F yang didefinisikan sebagai petunjuk pergerakan zero level set, tidak memiliki komponen untuk melestarikan fungsi level set phi sebagai *signed distance function*. Di dalam teori, hal ini dibuktikan oleh (Barles, et al., 1993) yang menunjukkan bahwa persamaan hamilton-jacobi tidak sesuai dengan *signed distance function*. Namun, fungsi level set bergerak menjadi *signed distance funcion* sebagai hasil dari reinisialisasi.

Hal ini tentu saja menjadi sebuah perbedaan pendapat antara teori dan implementasi, seperti yang dikemukakan pula pada Gomes dan Fauregas (Gomes & Fauregas, 2000). Dari sudut pandang praktik ini, penggunaan reinisialisasi mendatangkan masalah fundamental yang harus diselesaikan, seperti ketika kapan dan bagaimana menerapkan reinisialisasi (Gomes & Fauregas, 2000). Tidak ada jawaban yang mampu diterima secara umum untuk menghadapi masalah tersebut (Peng, et al., 1999) dan kemudian, proses reinisialisasi sering diterapkan secara *ad hoc*.

Berdasarkan latar belakang diatas (Li., et al., 2010) mengajukan formula level set baru yang disebut *distance regularization level set equation* (DRLSE) dimana formula ini menambahkan aturan regularisasi jarak dan aturan eksternal energi yang mengendalikan pergerakan kontur zero level set ke lokasi yang diinginkan. Aturan regulariasi jarak didefinisikan sebagai fungsi potensial yang mana bergerak pada gradient fungsi level set ke salah satu titik minimumnya, yang kemudian mempertahankan bentuk yang diinginkan pada fungsi level set, secara tepat sebuah signed distance mendekati *zero level set*.

Secara khas, kita memberikan fungsi potensial *double well* untuk regularisasi jarak.

Evolusi level set dirurunkan sebagai *gradient flow* yang meminimalisir energi fungsional. Di dalam evolusi level set, regularitas dari fungsi level set diatur oleh difusi *forward* dan *backward* yang diturunkan dari aturan regularisasi jarak. Sebagai hasilnya, regularisasi jarak secara penuh mengeliminasi kebutuhan reinisialisasi dan menghindari efek samping yang tidak diinginkan yang disebut sebagai aturan pinalti.

Secara umum *distance regularization level set evolution* memenuhi persamaan 9. Dimana turunan nilai phi terhadap waktu(t) merupakan akumulasi dari kekuatan eksternal, area dan regularisasi jarak terhadap gradien phi.

$$\frac{\partial \phi}{\partial t} = \mu \, \mathrm{div}(dp(|\nabla\phi|)\nabla\phi) + F|\nabla\phi| + A\nabla\phi \qquad (9)$$

Keterangan:

$\frac{\partial \phi}{\partial t}$	:	Turunan phi terhadap t
μ	:	Miu
$\mathrm{div}(dp(\lvert\nabla\phi\rvert)\nabla\phi)$	:	Kelengkungan Kontur Phi
$F\lvert\nabla\phi\rvert$	:	Gaya Phi
$A\nabla\phi$	:	Regulariasi Jarak Phi

Dengan *distance regularization term*, perhitungan numerik akan stabil tanpa harus melakukan reinisialisasi. Dalam segmentasi citra, sebelum melakukan perhitungan ini, terlebih dahulu perlu dilakukan penapisan derau menggunakan kernel gaussian yang bertujuan untuk mengurangi noise. Kemudian hitung fungsi energi dengan persamaan ke 10.

$$\epsilon(\phi) = \mu R_p(\phi) + \lambda \zeta_g(\phi) + a A_g(\phi) \quad (10)$$

Keterangan:

$\epsilon(\phi)$: Nilai Phi

μR_p : Akumulasi nilai miu dan Reinisialisasi phi

$\lambda \zeta_g(\phi)$: Kecepatan Rambat Kontur Phi

$a A_g(\phi)$: Luas Area Citra

4. Fuzzy Clustering

Dalam fuzzy clustering, titik pusat (*centroid*) dan daerah dari setiap subkelas diestimasi secara adaptif dalam rangka untuk meminimalisir *cost function*. Fuzzy c-means (FCM) merupakan salah satu dari algoritma yang paling populer dari *fuzzy clustering*, dan telah secara luas diaplikasikan di dalam masalah-masalah medis.

Algoritma FCM bersumber dari algoritma K means. Secara sederhana, algoritma k-means mencari sejumlah objek N, yang didasarkan dari atribut kedalam kluster-kluster K (K<N). Untuk segmentasi citra medis, N setara

dengan jumlah citra piksel Nx x Ny. Hasil yang diinginkan termasuk ke dalam *centroid* setiap kluster dan afiliasi dari objek N. *Clustering* k-means meminimalisir *cost function.*

$$J = \sum_{m=1}^{K} \sum_{n=1}^{N} ||i_n - v_m||^2 \quad (14)$$

Keterangan:

J : Nilai Titik Centroid

m : Jumlah Kluster

n : Indeks Data Piksel Citra

i : Data Piksel

v : Titik Centroid

Dimana i_n merupakan nilai piksel, v_m adalah titik centroid nilai ke m, dan ||.|| menandakan nilai absolut. Hasil ideal dari algoritma k-means adalah memaksimalkan variasi inter-cluster, namun meminimalisir intra-cluster. Didalam *clustering* k-means, setiap objek dibatasi oleh satu kluster K. Secara berkebalikan, algoritma FCM memanfaatkan sebuah derajat keanggotaan yang mengindikasikan derajat keanggotaan dari objek ke-n untuk cluster ke m, yang mana dijustifikasi untuk segmentasi citra medis sebagai jaringan fisik yang biasanya tidak homogen. *Cost function* di dalam FCM memenuhi persamaan 15.

$$J = \sum_{n=1}^{N} \sum_{m=1}^{C} \mu_{mn}^{l} ||i_n - v_m||^2 \quad (15)$$

Keterangan:

J : Nilai Titik Centroid

m : Jumlah Kluster

n : Indeks Data Piksel Citra

i : Data Piksel

v : Titik Centroid

μ : Nilai Miu Keanggotaan

Dimana nilai *l* merupakan sebuah parameter pengendali fuzzy dari resultan segmentasi yang nilainya lebih dari satu. Fungsi derajat keanggotaan memenuhi persamaan 16.

$$\sum_{m=1}^{c} \mu_{mn} = 1;\ 0 \leq \mu_{mn} \leq 1;\ \sum_{n=1}^{N} \mu_{mn} \geq 0 \quad (16)$$

Keterangan:

J : Nilai Titik Centroid

m : Jumlah Kluster

n : Indeks Data Piksel Citra

i : Data Piksel

v : Titik Centroid

μ : Nilai Miu Keanggotaan

Fungsi keanggotan dan titik pusat akan diperbaharui terus menerus dan algoritma FCM dioptimasikan ketika piksel-piksel yang berdekatan terhadap titik centroid memiliki nilai keanggotaan yang tinggi, sementara jika semakin jauh maka nilainya semakin rendah. Satu masalah utama dari algoritma FCM dalam segmentasi citra adalah kurangnya informasi spasial (Chuang, et al., 2006) (Cai, et al., 2007). Karena *noise* citra dan artifak sering merusak performa dari segmentasi FCM. (Cai, et al., 2007) mengajukan algoritma FCM yang tergeneralisasi untuk menggabungkan intensitas lokal dan informasi spasial. Hal ini memungkinkan pengggunaan operasi morfologi untuk mengaplikasikan batasan spasial ke langkah *postprocessing*. (Chuang et al., 2001) mengajukan algoritma FCM spasial yang mana informasi spasial dapat digabungkan ke dalam derajat keanggotaan fuzzy secara langsung menggunakan persamaan 17.

$$\mu_{mn} = \frac{\mu_{mn}^{p} h_{mn}^{q}}{\sum_{k=1}^{C} \mu_{kn}^{p} h_{kn}^{q}} \quad (17)$$

Keterangan:

μ_{mn}^{p} : Miu kluster m dan n

h_{mn}^{q} : Informasi spasial kluster m dan n

μ_{kn}^{p} : Miu kluster k dan n

h_{kn}^{q} : Informasi spasial k dan n

μ_{mn} : Miu kluster m dan n

Dimana *p* dan *q* adalah dua parameter pengendali. Variabel *h*, *m* dan *n* meliputi informasi spasial yang dapat digambarkan dengan persamaan 18.

$$h_{mn} = \sum_{k \in N_n} \mu_{nk} \quad (18)$$

Keterangan:

h_{mn} : Informasi Spasial Kluster m dan n

μ_{nk} : Miu n dan k

N_n : Daerah lokal yang terpusat antara n piksel citra

Dimana Nn merupakan sebuah daerah lokal yang terpusat antara n piksel citra. Sedangkan bobot miu dan titik pusat vm akan terus diperbaharui.

5. Fuzzy Level Set

Baik algoritma FCM dan level set, keduanya merupakan model komputasi *general-purpose* yang artinya dapat diimplementasikan pada segala masalah. Pada segmentasi citra medis, kita dapat mengambil keuntungan dari kedua metode tersebut untuk hasil yang lebih baik dari masing-masing algoritma. (Li, et al., 2011) mengajukan penggabungan algoritma tersebut untuk mengotomatisasi segmentasi citra medis. Algoritma ini dimulai dengan proses perhitungan *spatial fuzzy clustering*, yang mana hasil pengklusteran digunakan untuk menginisialisasi nilai fungsi level set, mengestimasi parameter pengendali dan meregularisasi evolusi level set.

Algoritma fuzzy level set mengotomatisasi nilai inisialisasi dan parameter pengendali dari segmentasi level set, menggunakan *spasial fuzzy clustering* yang menerapkan algoritma FCM dengan batasan spasial untuk menentukan aproksimasi nilai dari kontur *region of interest* citra medis. Keuntungan dari proses inisialisasi fleksibel ini ditunjukkan pada persamaan 8. Dimana fungsi level set dapat mengakomodasi hasil dari FCM secara langsung untuk evolusi. Pendukung komponen *region of interest* di dalam hasil FCM adalah R_k:$\{r_k = \mu_{nk}\ ,\ n = x.Ny + y\}$. Hal ini sangat cocok untuk diinisialisasikan ke fungsi level set sebagai zero level set yang memenuhi persamaan 19.

$$\phi_0(\mathrm{x},\mathrm{y}) = -4\varepsilon(0.5 - \mathrm{B_k}) \qquad (19)$$

Keterangan:

$\phi_0(x, y)$: Kontur Awal

-4ε : Konstanta -4 kali Epsilon

B_k : Daerah Citra Biner

Dimana ε merupakan konstanta pengendali dari fungsi *dirac*. Fungsi *dirac* didefinisikan sebagai persamaan 20 dan 21.

$$\delta_\varepsilon(x) = 0 \text{ jika } |x| \geq \varepsilon \quad (20)$$

Keterangan:

$\delta_\varepsilon(x)$: Fungsi Dirac

$|x|$: Nilai Piksel Citra

ε : Nilai Epsilon

$$\delta_e(x) = \frac{1}{2\varepsilon}\left[1 + \cos\left(\frac{\pi x}{\varepsilon}\right)\right] \text{ jika } |x| \leq \varepsilon \quad (21)$$

Keterangan:

$\delta_\varepsilon(x)$: Fungsi Dirac

$|x|$: Nilai absolut dari piksel

ε : Nilai epsilon Citra

Bk adalah citra biner yang didapatkan dari $B_k = R_k > b_0$ dimana nilai $b_0(\varepsilon(0,1))$ adalah nilai ambang batas yang dapat disesuaikan (*threshold).*

Keuntungan dari *spatial fuzzy clustering*, Bk dapat diaproksimasi dengan komponen *region of interest*, yang siap disesuaikan dengan *b0*. Terdapat beberapa parameter pengendali yang diasosiasikan dalam metode level set yang ditunjukan pada tabel 2.2.

Tabel 2.2 Parameter Pengendali Level Set

Parameter	Significance
σ	Pengendali sebaran fungsi penapisan derau
C	Pengendali kekuatan gradien dari fungsi inisialisasi level set
ε	Parameter pengatur fungsi Dirac δ(ϕ)
μ	Bobot koefisien pinalti
λ	Koefisien panjang kontur
v	Kecepatan evolusi level set
τ	Timestep evolution
T	Maksimum jumlah iterasi

Sangat penting untuk mengkonfigurasi parameter pengendali secara tepat, yang fungsinya bervariasi antara satu dan lainnya. Hingga saat ini tidak ada beberapa aturan umum untuk mengatur nilai dari parameter-parameter. Sebagai contoh, telah diketahui bahwa nilai sigma yang lebih besar membuat

citra semakin *smooth*, namun mengorbankan detail dari suatu citra. Waktu yang lebih besar (t) mampu mempercepat proses evolusi level set, namun beresiko terhadap pencapaian batasan kurva. Lebih dari itu penting untuk memilih nilai v (kecepatan) menjadi positif jika nilai inisial phi_0 diluar dari *component of interest*. Sebagai tambahan dengan melakukan *trial* dan *error*, beberapa aturan telah dilakukan untuk melakukan segmentasi level set (Sethian, 1999). Sebagai contoh hasil dari *time step* dan koefisien pinalti harus lebih kecil daripada 0.25 untuk evolusi agar berjalan dengan stabil dan parameter C harus lebih besar dari 2ε. Hal ini juga ditemukan dalam penelitian (Li, et al., 2011) bahwa nilai λ yang lebih besar membuat kontur lebih *smooth* dan nilai v yang lebih besar mampu mengakselerasi evolusi level set.

Secara atraktif penentuan parameter pengendali secara adaptif untuk segmentasi citra medis dilakukan sebagai berikut, diberikan inisial fungsi level set ϕ_0 dari *spatial fuzzy clustering* pada persamaan ke 19. Selanjutnya kita dapat mengestimasi panjang l dan area α yang ditunjukan pada persamaan 22 dan 23.

$$\mathrm{l} = \int_{1} \delta(\phi_0)\mathrm{dx}\,\mathrm{dy} \tag{22}$$

$$\alpha = \int_{1} \mathrm{H}(\phi_0)\mathrm{dx}\,\mathrm{dy} \tag{23}$$

Keterangan:

l : Panjang Area

$\delta(\phi_0)\, dx\, dy$: Fungsi Dirac Inisial phi

$H(\phi_0) dx\, dy$: Nilai Heaviside dari Kontur Awal

α : Area Citra

dimana fungsi heaviside $H(\phi_0)$ memenuhi persamaan 24.

$$H(\phi_0) = 1 \text{ jika } (\phi_0) > 0 \text{ dan nilainya } 0 \text{ jika } (\phi_0) < 0 \quad (24)$$

Keterangan:

$H(\phi_0)$: Fungsi Heaviside

ϕ_0 : Kontur Awal

Kita dapat mengobservasi evolusi level set akan lebih cepat jika *component of interest* berukuran semakin besar. Kecepatan (*v*) memiliki dua peran di dalam evolusi level set. Pertama, menentukan arah pergerakan dari fungsi level set, jika nilainya positif maka level set akan mengerucut dan negatif untuk mengembang.

Kedua, semakin besar nilai *v* maka semakin cepat level set berevolusi. Dalam algoritma level set konvensional, parameter pengendali *v* sering disebut sebagai konstanta global. Hal ini memberikan keuntungan untuk membuat fungsi level set berevolusi lebih cepat jika ϕ masih jauh dari batasan tujuan. Sebaliknya, fungsi level set harus semakin lambat ketika nilai ϕ semakin mendekati batasan yang dituju.

Selain itu fungsi level set harus mengubah arah secara otomatis. Sementara untuk menginisialisasi melalui batasan yang dituju pada penelitian (Li, et al., 2011) menemukan bahwa inisial segmentasi citra, sebagai index kuantitatif, yang secara khas berguna untuk meregularisasi evolusi level set. Algoritma *fuzzy level set* mengambil derajat keanggotaan dari setiap citra piksel μ_k sebagai jarak spesifik komponen dari daerah R_k. Peningkatan tekanan yang diajukan disini untuk menarik atau mendorong antarmuka dinamis secara adaptif melalui *object of interest.*

$$G(R_k) = 1 - 2R_k \tag{25}$$

Keterangan:

$\mathrm{G(R_k)}$: Resultan Tekanan Evolusi Level Set

Resultan tekanan evolusi level set $G(R_k)$ $(\varepsilon[-1\ 1])$ adalah matriks sebuah variabel yang menarik atau mendorong tekanan pada setiap citra. Dengan kata lain, fungsi level set akan menjadi tertarik melalui objek dari *region of intereset* pada posisi awalnya. Kemudian persamaan evolusi level set memenuhi persamaan 26.

$$\varsigma(\mathrm{g}, \phi) = \lambda\delta(\phi)\mathrm{div}\left(\mathrm{g}\,\frac{\nabla\phi}{\nabla\phi}\right) + \mathrm{gG(R_k)}\delta(\phi) \tag{26}$$

Keterangan:

$\varsigma(\mathrm{g}, \phi)$: Evolusi Level Set

$\lambda\delta(\phi)\mathrm{div}\left(g\,\frac{\nabla\phi}{\nabla\phi}\right)$: Kelengkungan Level Set

$gG(R_k)\delta(\phi)$: Tekanan dan Kecepatan Level Set

Persamaan 26 menunjukkan beberapa keuntungan dimana tekanan evolusi level set sekarang dapat diturunkan dari *spatial fuzzy clustering* secara langsung. lebih dari itu evolusi level set sekarang diadaptasi ke jarak pada objek asli. Ketika mendekati objek, fungsi level set akan secara otomatis memperlambat evolusi dan akan menjadi sangat bergantung pada aturan penapisan derau. Sebagai tambahan keuntungan lainnya adalah fleksibitias untuk memilih iterasi evolusi yang lebih besar dari evolusi level set dalam rangka untuk menghindari kekurangan atau kelebihan perhitungan komputasi (Li, et al., 2011).

6. Partial Differential Equation (PDE) dan Metode Numerik

6.1. Klasifikasi PDE

Persamaan differensial parsial merupakan persamaan differensial yang mengandung lebih dari satu macam persamaan. Karena citra digital merupakan fungsi matematika intensitas yang tergantung pada posisi pixel 2-dimensi, maka bentuk umum PDE untuk citra dapat ditulis seperti pada persamaan 27.

$$\frac{\partial^2 I}{\partial x^2} + B\,\frac{\partial^2 I}{\partial x\,\partial y} + C\frac{\partial^2 I}{\partial y^2} + D\frac{\partial I}{\partial x} + E\frac{\partial I}{\partial y} + FI = G \qquad (27)$$

Dalam persamaan di atas $I\,x, y$ mewakili intensitas citra dan merupakan variabel tak-bebas, sedangkan variabel x, y adalah variabel bebas. Orde dari PDE adalah tingkat turunan tertinggi yang ada pada PDE tersebut, seperti yang ditulis pada persamaan 28.

$$\frac{\partial^2 \mathrm{I}}{\partial \mathrm{x}^2} + \frac{\partial^2 \mathrm{I}}{\partial \mathrm{y}^2} \tag{28}$$

PDE juga digolongkan menjadi 2, yaitu: PDE linier dan nonlinier. PDE disebut linier jika tidak ada suku yang mengandung perkalian variabel tak-bebas ataupun turunannya, jika ada suku yang mengandung suku demikian maka PDE tersebut disebut PDE non-linier. Pada umumnya PDE non-linier lebih sulit dicari solusinya dari pada PDE linier. Karena PDE sebetulnya mewakili suatu fenomena fisis yang beraneka ragam dan mempunyai sifat yang khusus pula, maka solusinya juga berkaitan erat dengan fenomena yang dimodelkannya. Oleh karena itu PDE perlu diklasifikasi sehingga prosedur pencarian solusinya menyesuaikan dengan sifat PDE. Klasifikasi tersebut berdasar persamaan 27 dan 28 adalah seperti digambarkan pada tabel 2.3 (Hoffman dan Chiang, 2000).

Tabel 2.3 Klasifikasi PDE

Klasifikasi	**Syarat**	**Contoh fenomena fisik**
Hiperbolik	$B^2 < 4AC$	Perambatan gelombang
Parabolik	$B^2 = 4AC$	Perpindahan panas konduksi transien

Elliptik	$B^2 > 4AC$	Elektrostatik

6.2. Metode Beda Hingga

Solusi eksak dari PDE tersebut di atas sangat sulit diperoleh secara analitik, bahkan pada banyak kasus di bidang rekayasa jawaban eksak tidak mungkin diperoleh. Untuk masa sekarang, meskipun solusi eksak tidak dapat diperoleh tetapi solusi pendekatan yang berakurasi tinggi dimungkinkan untuk diperoleh. Metode numerik menawarkan solusi pendekatan tersebut, dengan metode numerik, PDE dibuat dalam bentuk diskret sehingga berbentuk sistem persamaan yang dapat diselesaikan dengan komputer. Metode numerik yang cocok untuk pemecahan PDE model citra digital adalah metode Beda Hingga karena metode ini membuat diskretisasi ruang berbentuk grid yang besesuaian dengan posisi piksel pada citra digital.

6.2.1 Metode Beda Hingga Satu Variabel

Metode Beda Hingga dikembangkan berdasar Deret Taylor yang mengekspansikan suatu fungsi menjadi deret polinomial. Ekspansi Deret Taylor untuk suatu fungsi *f(x)* yang mempunyai 1 variabel di sekitar nilai *x* digambarkan pada persamaan 29.

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2!} f''(x) + \frac{h^3}{3!} f'''(x) + O(h^4) \qquad (29)$$

Notasi $f'(x) = \frac{df}{dx}(x)$ menyatakan turunan pertama, $O(h^4)$ menunjukkan bahwa deret dipotong sampai suku ketiga kesalahan perhitungan atau galat sebanding dengan h^4. Pendekatan turunan pertama dari fungsi *f(x)* diperoleh dengan memotong deret sampai suku kedua sehingga kesalahan perhitungan sebanding dengan h^2.

$$f'(x) = \frac{f(x+h) - f(x)}{h} + O(h^1) \tag{30}$$

Dalam perhitungan turunan tersebut kesalahan perhitungan O(h^2) dibagi dengan h sehingga order kesalahan perhitungan turunan pertama sebanding dengan O(h^1). Karena pendekatan turunan pada titik x menggunakan nilai fungsi di titik x dan *x+h*, maka skema ini disebut Beda Maju (Forward Difference). Penggunaan ekspansi, arah mundur yaitu arah negatif juga dapat dilakukan sehingga skema yang dihasilkan disebut skema beda mundur yang ditunjukkan pada persamaan 31.

$$f'(x) = \frac{f(x) - f(x-h)}{h} + O(h)^1 \tag{31}$$

Dengan mengkombinasikan skema Beda Maju dan Beda Munjur yang ditunjukkan pada persamaan 32-35 diperoleh skema beda tengah (***central difference***) yang mengandung kesalahan berorde O(h^2). Kombinasi Skema Beda Maju dan Mundur dilakukan dengan mengurangi persamaan 30 dan 31.

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2!} + f''(x) + O(h^3) \tag{32}$$

$$f(x-h) = f(x) - hf'(x) + \frac{h^2}{2!} + f''(x) + O(h^3) \quad (33)$$

$$f(x+h) - f(x-h) = 2hf'(x) \quad (34)$$

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} \quad (35)$$

Untuk melakukan pendekatan perhitungan turunan kedua yang mempunyai kesalahan berorde h^2 diperoleh dengan mempertahankan suku order ke h^3. Nilai-nilai di atas kemudian dimasukkan ke dalam perhitungan metode Beda Hingga, Pada metode ini maka diperoleh hasil perhitungan seperti tertulis pada persamaan 36-39.

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2!}f''(x) + \frac{h^3}{3!}f'''(x) + O(h^4) \quad (36)$$

$$f(x-h) = f(x) - hf'(x) + \frac{h^2}{2!}f''(x) - \frac{h^3}{3!}f'''(x) + O(h^4) \quad (37)$$

$$h^2f''(x) = f(x+h) - 2f(x) + f(x-h) + O(h^4) \quad (38)$$

$$f''(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} + O(h^2) \quad (39)$$

Tabel 2.4 Skema PDE

Skema Beda Maju	$\frac{df}{dx} = \frac{f_{i+1} - f_i}{h}$
Skema Beda Mundur	$\frac{df}{dx} = \frac{f_i - f_{i-1}}{h}$

Skema Beda Tengah	$\frac{df}{dx} = \frac{f_{i+1} - f_{i-1}}{2h}$
Skema Turunan Kedua	$\frac{d^2f}{dx^2} = \frac{f_{i+1} - 2f_i + f_{i-1}}{h^2}$

6.2.2. Metode Beda Hingga Dua Variabel

Pendekatan numerik Beda Hingga untuk turunan fungsi multivariabel akan dijelaskan pada bagian berikut ini. Fungsi multivariabel minimal mempunyai 2 variabel tak bebas, sehingga pendekatan turunannya juga minimal mempunyai dimensi dua. Seperti halnya fungsi dengan 1 variabel, maka pendekatan turunan multivariabel juga menggunakan Deret Taylor. Ekspansi Deret Taylor untuk fungsi *f* di dekat titik (x,y) memenuhi persamaan 40.

$$f(x+h, y+k) = f(x,y) + h\frac{\partial f}{\partial x}(x,y) + k\frac{\partial f}{\partial y}(x,y) + \frac{h^2}{2!}\frac{\partial^2 f}{\partial x^2}(x,y) + \frac{2hk}{2!}\frac{\partial^2 f}{\partial x\,\partial y}(x,y) + \frac{k^2}{2!}\frac{\partial^2 f}{\partial y^2}(x,y) + O(h^3) \quad (40)$$

Konstanta h dan k adalah suatu konstanta yang berilai kecil. Ekspansi pada persamaan 40 dapat juga ditulis dengan persamaan vektor yang ditunjukkan pada persamaan 41. Adapun penjabaran detail dari persamaan ditunjukkan pada persamaan 42.

$$f(x+h) = f(x) + h^T.\nabla f(x) + \frac{1}{2!}h^T.\nabla\nabla f(x).h + O(|h|^3) \quad (41)$$

$$h = \begin{bmatrix} h \\ k \end{bmatrix}, h^T = (h\ k), x = \begin{bmatrix} x \\ y \end{bmatrix}, \nabla\nabla = \frac{\partial^2}{\partial x\, \partial y} \quad (42)$$

Notasi vektor di atas berguna jika analisis numerik melibatkan suku sisa dengan turunan tinggi, tetapi karena pendekatan PDE untuk pengolahan citra tidak memerlukan turunan berorde tinggi maka notasi vektor tidak digunakan dalam penelitian ini. Ekspansi Deret Taylor fungsi f(x,y) pada sumbu-x *direction* memenuhi persamaan 43 dan 44.

$$f(x+h,y) = f(x,y) + h\frac{\partial f}{\partial x}(x,y) + \frac{h}{2!}(x,y)\frac{\partial f}{\partial x^2}(x,y) + O(h^3) \quad (43)$$

$$f(x-h,y) = f(x,y) - h\frac{\partial f}{\partial x}(x,y) + \frac{h^2}{2!}\frac{\partial^2 f}{\partial x^2}(x,y) + O(h^3) \quad (44)$$

$$\frac{\partial f}{\partial x}(x,y) = f_x(x,y) = \frac{f(x+h,y) - f(x,y)}{2h} + O(h^1) \quad (45)$$

$$\frac{\partial f}{\partial x}(x,y) = f_x(x,y) = \frac{f(x,y) - f(x-h,y)}{2h} + O(h^1) \quad (46)$$

$$\frac{\partial f}{\partial x}(x,y) = f_x(x,y) = \frac{f(x+h,y) - f(x-h,y)}{2h} + O(h^2) \quad (47)$$

$$\frac{\partial^2 f}{\partial x^2}(x,y) = \frac{f(x+h,y) - 2f(x+h,y) + f(x-h,y)}{h^2} + O(h^2) \quad (48)$$

	f(x,y+k) i,j+1	

f(x-h,y) i-1,j	f(x,y) i,j	f(x+h,y) i+1,j
	f(x,y-k) i,j-1	

Gambar 2.1. Skema Beda Hingga 2 Variabel

Turunan parsial terhadap variabel *y* dapat dicari dengan cara yang sama dengan skema pada gambar 2.1. Skema beda hingga 2 variabel apabila dinyatakan dengan notasi diskret ditunjukkan pada tabel 2.5. Skema beda maju pada sumbu horizontal dituliskan dengan indeks i, skema ini didapatkan dengan selisih dari indeks sumbu i ditambah satu dan indeks sumbu i saat ini dibagi dengan nilai h. Skema beda maju pada koordinat vertikal yang dituliskan dengan notasi indeks j didapatkan dengan selisih dari indeks sumbu j ditambah satu dan indeks sumbu j saat ini dibagi dengan nilai h. Pada skema beda mundur sumbu horizontal didapatkan dengan mendapatkan selisih nilai indeks i saat ini dan i dikurangi 1 dibagi h. Pada skema beda mundur sumbu vertikal didapatkan dengan mendapatkan selisih nilai indeks j saat ini dan j dikurangi 1 dibagi h. Hasil dari penggabungan skema beda maju dan beda mundur pada ruang diskretisasi menghasilkan skema beda tengah.

Tabel 2.5 Skema Turunan Dua Variabel

Beda Maju	$\frac{\partial f}{\partial x} = \frac{f_{i+1,j} - f_{i,j}}{h}$	$\frac{\partial f}{\partial y} = \frac{f_{i,j+1} - f_{i}}{h}$
Beda Tengah	$\frac{\partial f}{\partial x} = \frac{f_{i+1,j} - f_{i-1,j}}{2h}$	$\frac{\partial f}{\partial y} = \frac{f_{i,j+1} - f_{i,j-1}}{2h}$
Beda Mundur	$\frac{\partial f}{\partial x} = \frac{f_{i,j} - f_{i-1,j}}{h}$	$\frac{\partial f}{\partial y} = \frac{f_{i,j} - f_{i,j-1}}{h}$
Turunan Kedua	$\frac{\partial^2 f}{\partial^2 x} = \frac{f_{i+1,j} - 2f_{i,j} + f_{i-j,j}}{h^2}$	$\frac{\partial^2 f}{\partial^2 x} = \frac{f_{i,j+1} - 2f_{i,j} + f_{i,j-1}}{h^2}$

7. Parallel CUDA

Sejak tahun 2003, ada dua perkembangan utama pada pembuatan *mikroprocessor* yaitu *multicore* dan *manycore* (Kirk & Hwu, 2010). *Multicore* mulai dengan prosesor berinti dua yang kemudian berkembang sampai ke prosesor berinti 8. Tujuan *multicore* adalah mempertahankan kinerja untuk menjalankan program sekuensial.

Multicore dapat dilihat pada bentuk CPU. Sedangkan *manycore* terfokus pada aplikasi parallel. *Manycore* lebih banyak memiliki unit pemroses dibandingkan *multicore*. *Manycore* banyak dijumpai pada bentuk GPU. Perbedaan yang mendasar antara CPU dan GPU didasarkan pada dasar desain kedua tipe *processor*. CPU dibuat untuk mengoptimalkan program sekuensial sedangkan GPU dibuat untuk mengoptimalkan program parallel. Hal yang lain yang mendasari perbedaan ini adalah jumlah *bandwidth* pada memory. GPU mempunyai *bandwidth* 10 kali dibandingkan CPU. Tetapi pada dasarnya GPU tidak dapat menggantikan CPU, karena GPU hanya untuk membuat komputasi parallel untuk program parallel dan bukan untuk program sekuensial.

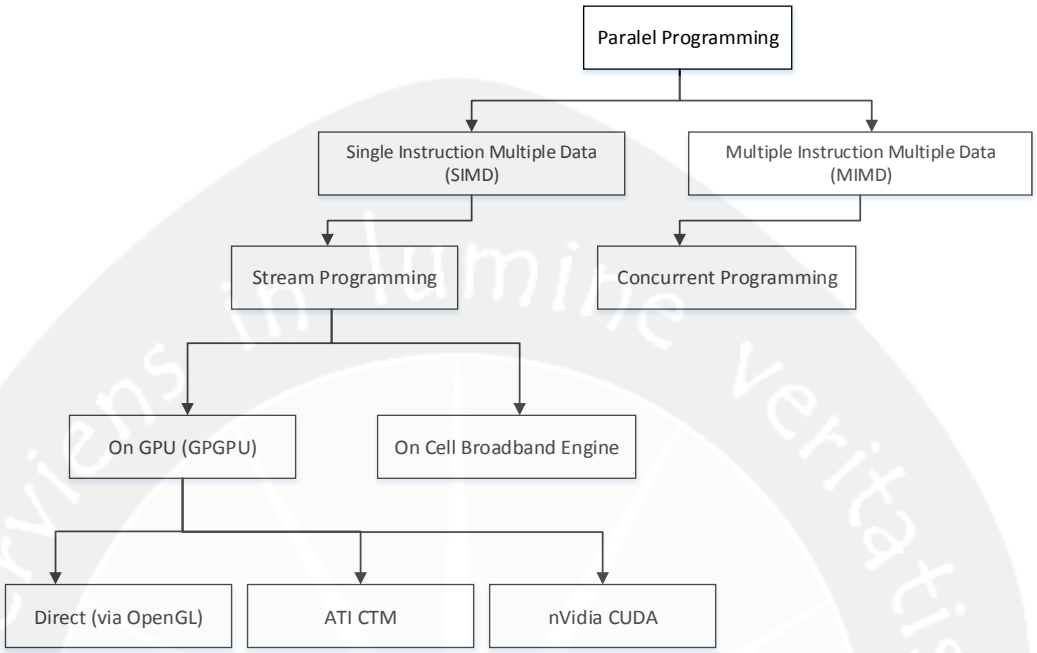
7.1 Parallel computing

Komputasi paralel merupakan salah satu teknik untuk melakukan komputasi secara bersamaan dengan memanfaatkan beberapa komputer. Pada komputasi paralel dibutuhkan kapasitas yang sangat besar untuk memproses

komputasi yang banyak, disamping itu pemakai harus menerapkan bahasa pemrograman paralel untuk dapat merealisasikan komputasi. (Kirk & Hwu, 2010).

Pemrograman paralel memiliki tujuan utama yaitu untuk meningkatkan performa komputasi karena semakin banyak hal yang bisa dilakukan secara bersamaan dalam waktu yang sama, maka semakin banyak pula pekerjaan yang bisa diselesaikan (Kirk & Hwu, 2010).

Stream Programming adalah sebuah teknik pemrograman konkruen yang dapat digunakan untuk melakukan komputasi paralel. Dalam *stream programming*, data diorganisasi menjadi sekumpulan *stream*, dan sebuah program yang disebut *kernel* diaplikasikan pada setiap elemen *stream.* Dari gambar 2.1 dapat dilihat bahwa s*tream programming* menggunakan paradigma *Single Instruction Multiple Data* (SIMD). Artinya, program yang sama dieksekusi pada sekumpulan data secara paralel. (Kirk & Hwu, 2010)



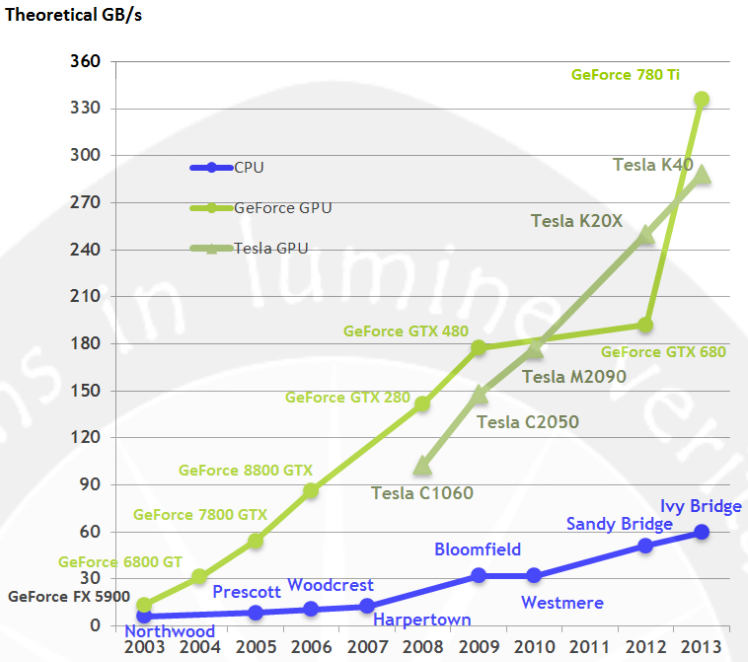
Gambar 2.2. Stream Programming dalam Taksonomi Parallel Programming

Salah satu perangkat keras yang dapat digunakan untuk melakukan stream programming adalah *Graphics Processing Unit* (*GPU*). GPU sangat cocok untuk diprogram dengan teknik ini, karena GPU dirancang dari awal untuk melakukan proses *rendering* 3D, dimana proses tersebut membutuhkan *hardware* yang dirancang secara khusus untuk memproses data secara SIMD. Untuk mengimplementasikan stream programming pada GPU, ada banyak framework yang dapat dipakai. Salah satunya adalah *Compute Unified Device Architecture* (CUDA). (CUDA™, 2012)

7.2 Kemampuan Komputasi GPU

Graphic Processing Unit (GPU) adalah salah satu *peripheral* komputer yang bertugas untuk melakukan proses *rendering* objek 3D ke layar. Karena *rendering* merupakan proses yang sangat paralel, maka evolusi arsitektur GPU mengikuti perkembangannya: GPU berevolusi menjadi sebuah multiprosesor yang mampu menjalankan tugas secara paralel.

GPU dapat disebut unit komputasi yang terpisah dari CPU karena GPU memiliki prosesor dan memori sendiri. GPU memiliki jumlah *Arithmetic Logic Unit* (ALU) yang jauh lebih besar dibandingkan komputer biasa. Oleh karena itu, kemampuan GPU dalam mengolah data yang besar dapat diandalkan. Pada gambar 2.2, dapat dilihat bahwa pertumbuhan kemampuan GPU jika diukur dengan *Floating Point Operations per Second* (FLOPS) jauh meninggalkan CPU, bahkan meninggalkan hukum Moore.



Gambar 2.3. Peningkatan Kecepatan GPU

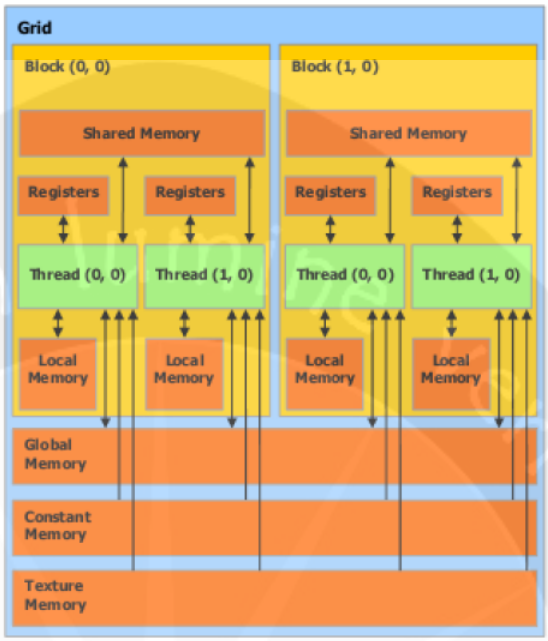
7.3 Compute Unified Device Architecture (CUDA)

CUDA™ adalah *platform* komputasi paralel dan model pemrograman yang memungkinkan peningkatan dramatis dalam kinerja komputasi dengan memanfaatkan kekuatan dari *graphics processing unit* (*GPU*). (Kirk & Hwu, 2010).

Sejak diperkenalkan pada 2006, CUDA telah banyak disebarkan melalui ribuan aplikasi dan diterbitkan makalah penelitian, dan terinstal lebih dari 300 juta CUDA-*enabled GPU* di *notebook*, *workstation*, komputer *cluster* dan super-komputer. *GPU* diaplikasikan juga untuk astronomi, biologi, kimia, fisika, *data mining*, manufaktur, keuangan. (CUDA™, 2012).

Pada gambar 2.3 menerangkan tentang hirarki memori dalam CUDA. Setiap *stream processor*, yang dianalogikan dengan *thread*(x,y) dikelompokan dalam *block block*. Setiap *block* memiliki memori bersama yang dinamakan *shared memory*. *Shared memory* adalah *memory* berukuran kecil berkecepatan tinggi yang hanya dapat diakses *stream processor* dalam *block* yang sama. *Bandwidth* antara *stream processor* dan *shared memory* sangat besar, mencapai 70,7 GB/s. Jenis memory lain adalah *device memory*. *Memory* ini dapat diakses oleh seluruh *multiprocessor*, namun dengan *bandwidth* yang lebih lambat. Adapun *memory* komputer diistilahkan dengan *host memory*. Implementasi konsep stream programming pada CUDA sebagai berikut :

1. *Stream*: Dalam CUDA, *stream* disimpan dalam sebuah memori milik GPU yang bernama *global memory*. *Stream* diorganisasi berdasarkan jumlah *thread* yang akan mengaksesnya secara paralel. Setelah itu, *thread* tersebut harus diorganisasikan menjadi *thread block*.
2. *Kernel*: Dalam CUDA, *kernel* tetap merupakan sebuah program yang beroperasi pada data yang terdapat pada *global memory*.



Gambar 2.4. Hirarki Memori Pada CUDA

Langkah-langkah umum untuk melakukan komputasi pada CUDA adalah sebagai berikut:

1. Tentukan jumlah *thread* yang akan beroperasi pada *stream*. Kemudian tentukan jumlah *thread per block* dan jumlah *thread block* yang efisien pada hardware CUDA. Jumlah tersebut dinamakan *execution configuration*.
2. Pindahkan semua data dari *host memory* ke *device memory*. Ini harus dilakukan karena GPU tidak bisa membaca *host memory*.
3. Lakukan komputasi pada *device memory* beserta *execution configuration* nya. Data dalam *device memory* ini akan menjadi *stream*.
4. Pindahkan hasil komputasi pada *device memory* ke *host memory*