Grid-based General Type-2 Fuzzy Logic Systems based on GPU Computing

Long Thanh Ngo
Department of Information Systems
Le Quy Don Technical University, Hanoi, Vietnam
Email: ngotlong@gmail.com

***Abstract*—The advances of nVIDIAs computing technology allow to deploy of general purpose high performance applications of Graphic Processing Units (GPU) by parallelizing algorithms. Based on the grid-based representation of T2FS, the paper deals with an approach to computational process of general type-2 fuzzy logic systems based on Grid T2FS for GPU platforms. Experiments are implemented in various applications to show that the representation are suitable to speed-up general type-2 fuzzy logic systems on GPU platform.**

***Index Terms*—Type-2 fuzzy set, General type-2 fuzzy logic systems, Graphic Processing Units, defuzzification, Type-2 fuzzy operation.**

I. Introduction

Nowadays, many researches focus on representation of type-2 fuzzy sets for reducing complexity of computation of operations, especially for type-2 fuzzy logic systems. Because of advantages of type-2 fuzzy sets in management of uncertainty, applications of type-2 fuzzy sets are deployed in many fields. Almost applications are limited by computational complexity of general type-2 fuzzy sets. Recently, these are many researches arising from reduction the complexity of these systems that depend on representation of type-2 fuzzy sets. There are many approaches to representation of type-2 fuzzy sets. The point-based representation [5], [6] has proposed with operations and computation of type-2 fuzzy sets and systems. The point-based pure representation of type-2 fuzzy sets are hardly to apply because of huge cloud of points by discretizing the domain. Mendel et al [6] proposed the representation based on embedded fuzzy sets, vertical slices and gains successful from approximate computation of embedded type-2 fuzzy sets for type-2 fuzzy logic systems. Starczewski [10] introduced a method for complexity reduction of operations on triangular type-2 fuzzy sets. For this purpose, Coupland et al [2] proposed geometric method for representation type-1 and interval type-2 fuzzy sets, new algorithms for various operations on type-1 and type-2 fuzzy sets and for defuzzification. Coupland et al [3], [4] presented new techniques using upper and lower surfaces for performing logical operations on type-2 fuzzy sets with considering computational speed and accuracy. Techniques based on cloud of triangles are limited by computational capability in case of complexity type-2 fuzzy sets with huge quantity of triangles.

Triangulated irregular network (TIN) has used to represent 3-dimensional surfaces by dividing domain into irregular sub-triangles and approximately represent using piecewise linear functions. To support speed-up in computation of operations of type-2 fuzzy sets, a refinement CTIN [9] is proposed by combining contour model and constraint Delaunay criteria. So a type-2 fuzzy set is also represented by RCTIN in the same way. An approach was proposed to use RCTIN for representation of general type-2 fuzzy sets, algorithms for operations and general type-2 fuzzy logic systems (GT2FLS).

Hagras et al [11] proposed an approach to representation of type-2 fuzzy sets using zSlices that be able to gap interval type-2 fuzzy sets and general type-2 fuzzy sets. A similar approach to representation of type-2 fuzzy sets was proposed by Mendel et al [7] based on α - plane representation.

CUDA-based GPU computing has developed by nVIDIA and gains advantages with capability to parallel algorithms into sub-algorithms on threads, blocks as matrices. General purpose GPU has applied for intelligent computation such as fuzzy c-mean clustering [12], fuzzy neural networks [16], fuzzy ART clustering [15] or fuzzy logic systems [13], [14].

On the basis of matrix-based parallelisation, grid type-2 fuzzy sets are proposed to take this advantage. The paper introduces an approach to representation of grid general type-2 fuzzy sets, that use a grid to store data at vertices as matrices and value of non-vertex points is interpolated based on four vertices of rectangle containing them. This representation is the basis to design algorithms of general type-2 fuzzy sets on GPU paralleling platform. Computational process of general type-2 fuzzy logic systems based on Grid T2FS are fully proposed on both of platforms GPU and CPU that points out the advantage of GPU-based high performance in comparison with CPU platform. Experiments are implemented on GPU and CPU platforms with summarised reports on various input parameters such as resolution of grid to show advantages of this approach. An application of grid type-2 fuzzy logic systems are given based on avoidance behavior of robot navigation. Runtime of algorithms on GPU is enough fast to deploy applications of type-2 fuzzy sets to real problems with high accuracy.

The paper is organized as follows: II presents an overview on type-2 fuzzy sets and GPU computing; III introduces grid type-2 fuzzy sets, operations involving join, meet, negation and defuzzification; IV presents experiments on representation and operations with various parameters of input grid type-2 fuzzy sets; V is conclusion and future works.

II. Type-2 Fuzzy Logic Systems and GPU Computing

A. Type-2 Fuzzy Logic Systems

A type-2 fuzzy set in X is denoted $\tilde{A}$, and its membership grade of $x \in X$ is $\mu_{\tilde{A}}(x,u), u \in J_x \subseteq [0,1]$, which is a type-1 fuzzy set in [0, 1]. The elements of domain of $\mu_{\tilde{A}}(x,u)$ are called primary memberships of x in $\tilde{A}$ and memberships of primary memberships in $\mu_{\tilde{A}}(x,u)$ are called secondary memberships of x in $\tilde{A}$.

Definition 2.1: A $type-2\ fuzzy\ set$, denoted $\tilde{A}$, is characterized by a type-2 membership function $\mu_{\tilde{A}}(x,u)$ where $x \in X$ and $u \in J_x \subseteq [0,1]$, i.e.,

$$\tilde{A} = \{((x,u), \mu_{\tilde{A}}(x,u)) | \forall x \in X, \forall u \in J_x \subseteq [0,1]\} \quad (1)$$

or

$$\tilde{A} = \int_{x \in X} \int_{u \in J_x} \mu_{\tilde{A}}(x,u))/(x,u), J_x \subseteq [0,1] \quad (2)$$

in which $0 \leq \mu_{\tilde{A}}(x,u) \leq 1$.

At each value of x, say $x = x'$, the 2-D plane whose axes are u and $\mu_{\tilde{A}}(x',u)$ is called a $vertical\ slice$ of $\mu_{\tilde{A}}(x,u)$. A *secondary membership function* is a vertical slice of $\mu_{\tilde{A}}(x,u)$. It is $\mu_{\tilde{A}}(x = x', u)$ for $x \in X$ and $\forall u \in J_{x'} \subseteq [0,1]$, i.e.

$$\mu_{\tilde{A}}(x = x', u) \equiv \mu_{\tilde{A}}(x') = \int_{u \in J_{x'}} f_{x'}(u)/u, J_{x'} \subseteq [0,1] \quad (3)$$

in which $0 \leq f_{x'}(u) \leq 1$.

Let $\tilde{A}, \tilde{B}$ be type-2 fuzzy sets whose secondary membership grades are $f_x(u), g_x(w)$, respectively. Theoretic operations of type-2 fuzzy sets such as union, intersection and complement are described [5] as follows:

$$\mu_{\tilde{A} \cup \tilde{B}}(x) = \mu_{\tilde{A}}(x) \sqcup \mu_{\tilde{B}}(x) = \int_u \int_v (f_x(u) \star g_x(w))/(u \vee w) \quad (4)$$

$$\mu_{\tilde{A} \cap \tilde{B}}(x) = \mu_{\tilde{A}}(x) \sqcap \mu_{\tilde{B}}(x) = \int_u \int_v (f_x(u) \star g_x(w))/(u \star w) \quad (5)$$

$$\mu_{\tilde{A}}(x) = \mu_{\neg \tilde{A}}(x) = \int_u (f_x(u))/(1-u) \quad (6)$$

where $\vee, \star$ are t-cornorm, t-norm, respectively. Type-2 fuzzy sets are called an interval type-2 fuzzy sets if the secondary membership function $f_{x'}(u) = 1\ \forall u \in J_x$.

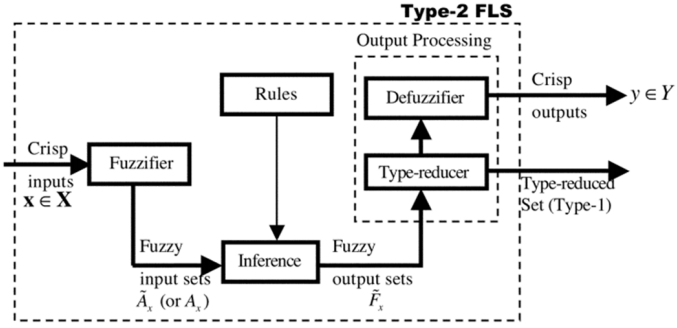

Fig. 1. The structure of type-2 fuzzy logic system

Fig. 1 shows the structure of a type-2 FLS. The singleton fuzzifier maps the crisp input into a type-2 fuzzy set. Singleton fuzzification is only considered, for which the input fuzzy set has only a single point of nonzero membership. "IF-THEN" rules that the l^{th} rule has the form "R^l :

IF x_1 is $\tilde{F}_1^l$ and x_2 is $\tilde{F}_2^l$ and ... and x_p is $\tilde{F}_p^l$ THEN y is $\tilde{G}^l$",

where: x_is are inputs; $\tilde{F}_i^l$s are antecedent sets (i=1, 2, ..., p); $\tilde{G}^l$ is the output.

The output of the inference engine is a type-2 set. Type-reduction is used to take from the type-2 output sets of the FLS to a type-1 set. To obtain a crisp output from a type-2 FLS, we can defuzzify the type-reduced set and the most natural method is by finding the centroid of the type-reduced set.

The following is the inference processing [4] of type-2 FLS: Consider a type-2 FLS having p inputs, $x_1 \in X_1, x_2 \in X_2, ..., x_p \in X_p$, and one output $y \in Y$. Let us suppose that it has M rules where the l^{th} rule has the form

$$R^l : \text{IF } x_1 \text{ is } \tilde{F}_1^l \text{ AND ... AND } x_p \text{ is } \tilde{F}_p^l \text{ THEN } y \text{ is } \tilde{G}^l. \quad (7)$$

This rule represents a type-2 fuzzy relation between the input space $X_1 \times X_2 \times ... \times X_p$ and the output space Y of the FLS. We denote the membership function of this type-2 relation as $\mu_{\tilde{F}_1^l \times ... \times \tilde{F}_p^l \to \tilde{G}^l}(x,y)$, where $\tilde{F}_1^l \times ... \times \tilde{F}_p^l$ denotes the Cartesian product of $\tilde{F}_1^l$, ..., $\tilde{F}_p^l$, and $x = \{x_1, x_2, ..., x_p\}$.

When an input $\mathbf{x}'$ is applied, the composition of the fuzzy set $\tilde{X}'$ to which $\mathbf{x}''$ belongs and the rule R^l is found by using the extended sup-star composition

$$\mu_{\tilde{X}' \circ \tilde{F}_1^l \times ... \times \tilde{F}_p^l \to \tilde{G}^l}(y) = \sqcup_{x \in \tilde{X}'} [\mu_{\tilde{X}'}(x) \sqcup \mu_{\tilde{F}_1^l \times ... \times \tilde{F}_p^l \to \tilde{G}^l}(x,y)] \quad (8)$$

We denote $\tilde{X}' \circ \tilde{F}_1^l \times ... \times \tilde{F}_p^l \to \tilde{G}^l$ as $\tilde{B}^l$, the output set corresponding to the l^{th} rule. We use singleton fuzzification and the product or minimum implication, inference process of rule l^{th} is described (detail in [4]) as follows:

$$\begin{aligned} \mu_{\tilde{B}^l}(y) &= \mu_{\tilde{F}_1^l}(x_1) \sqcap \mu_{\tilde{F}_2^l}(x_2) \sqcap ... \sqcap \mu_{\tilde{F}_p^l}(x_p) \sqcap \mu_{\tilde{G}^l}(y) \\ &= \mu_{\tilde{G}^l}(y) \sqcap [\sqcap_{i=1}^p \mu_{\tilde{F}_i^l}(x_i)] \end{aligned} \quad (9)$$

The output sets of rules are combined in a type-2 fuzzy set by using $join$ operation.

B. GPU Computing

Recently, the GPU has been transformed into the general purpose GPU. The programmable GPU has evolved into a high parallel, multi-threaded, multi-core processor with huge computational power and memory bandwidth. Fig. 2 shows CUDA processing model. CUDA allows multiple kernels to be run simultaneously on a single GPU. CUDA refers to each kernel as a grid. A grid is a collection of blocks. Each block runs the same kernel but is independent of each other. A block contains threads, which are the smallest divisible unit on a GPU.

The CUDA programming model is a set of massive threads that run in parallel. A thread block is a number of SIMD(Single Instruction, Multiple Data) threads that work on an Streaming Multiprocessor at a given time, can exchange information through the shared memory, and can be synchronized. The operations are systematized as a grid of thread

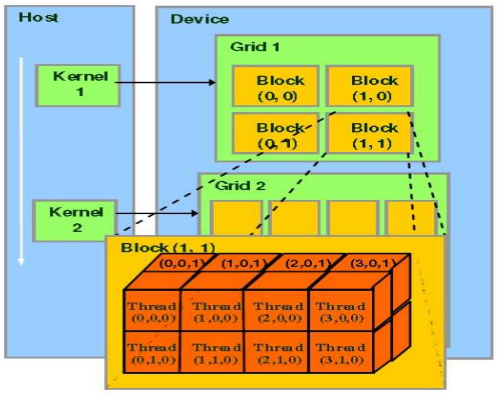

Fig. 2. CUDA processing model design

blocks. For operation parallelism, the programming model allows a developer to partition a program into several sub-problems, each of which is executed independently on a block. Each sub-program can be further divided into finer pieces that perform the same function for execution on different threads within the block. For dataset parallelism, datasets can be divided into smaller chunks that are stored in the shared memory, and each chunk is visible to all threads of the same block. This local data arrangement approach reduces the need to access off-chip global memory, which reduces data access time.

C. Grid Type-2 Fuzzy Sets

Grid type-2 fuzzy sets and operations have proposed [20] to take the advantages of high performance computing on GPU platform. This section reviews concepts and operations of grid type-2 fuzzy sets. Let X be domain of type-2 fuzzy set $\tilde{A}$ and $U = (u_{min}, u_{max}) \subseteq [0,1]$ be secondary domain of $\tilde{A}$. Space $X \times U$ of $\tilde{A}$ can be divided into a grid with union of $M \times N$ cells. A sub type-2 fuzzy set $\tilde{A}_{ij}$ in domain of the cell (i,j) is described as follows:

$$\tilde{A}_{ij} = \{((x,u), \mu_{\tilde{A}}(x,u) | x \in X_i = [\underline{x}_i, \overline{x}_i], u \in U_j = [\underline{u}_j, \overline{u}_j]\} \quad (10)$$

in which $\underline{x}_i = x_{min} + i * dx$, $\overline{x}_i = \underline{x}_i + dx$, $\underline{u}_j = u_{min} + j * du$, $\overline{u}_j = \underline{u}_j + du$, $i = \overline{0, N-1}$ and $j = \overline{0, M-1}$.

To define grid type-2 fuzzy set, called $\tilde{A}^g$, an approximate representation of sub type-2 fuzzy set $\tilde{A}_{ij}$ by a cell type-2 fuzzy set $\tilde{A}^c_{ij}$ is introduced. Let $f_{ij}(x,u)$ be to compute membership grade of $\tilde{A}^c_{ij}$ at (x,u) and is described as follows:

Provide that (x,u) is in the cell (i,j) and $d_k (k = \overline{1,4})$ are distance from (x,u) to k^{th} vertex of the cell, i.e.

$$\begin{aligned} d_1 &= \sqrt{(x-\underline{x})^2 + (u-\underline{u})^2} \\ d_2 &= \sqrt{(x-\underline{x})^2 + (u-\overline{u})^2} \\ d_3 &= \sqrt{(x-\overline{x})^2 + (u-\overline{u})^2} \\ d_4 &= \sqrt{(x-\overline{x})^2 + (u-\underline{u})^2} \end{aligned} \quad (11)$$

Let u_k be the rate between (x_u) and k^{th} vertex and computed as follows:

$$u_k = \frac{1}{\sum_{i=1}^{4} \sqrt{d_k / d_i}} \quad (12)$$

And the membership grade at (x,u) of $\tilde{A}$ is computed as follows:

$$f_{ij}(x,u) = \sum_{i=1}^{4} u_i \times f^{(i)} / \sum_{i=1}^{4} u_i \quad (13)$$

in which $f^{(1)} = \mu_{\tilde{A}_{ij}}(\underline{x}, \underline{u})$, $f^{(2)} = \mu_{\tilde{A}_{ij}}(\underline{x}, \overline{u})$, $f^{(3)} = \mu_{\tilde{A}_{ij}}(\overline{x}$, $f^{(4)} = \mu_{\tilde{A}_{ij}}(\overline{x}, \underline{u})$.

Definition 2.2: A cell type-2 fuzzy set, denoted $\tilde{A}^c_{ij}$, is approximate representation of sub type-2 fuzzy set $\tilde{A}_{ij}$ and is defined as follows:

$$\tilde{A}^c_{ij} = \{((x,u), \mu_{\tilde{A}^c_{ij}}(x,u)) | x \in [\underline{x}_i, \overline{x}_i], u \in [\underline{u}_i, \overline{u}_i]\} \quad (14)$$

in which $\mu_{\tilde{A}^c_{ij}}(x,u)) = f_{ij}(x,u)$ if $(x,u) \in D_{ij}$ and D_{ij} is domain of the cell (i,j), $i = \overline{0, N-1}$, $j = \overline{0, M-1}$.

Definition 2.3: A grid type-2 fuzzy set, denoted $\tilde{A}^g$, is union of above defined cell type-2 fuzzy set, i.e

$$\tilde{A}^g = \bigcup_{i=0}^{N-1} \bigcup_{j=0}^{M-1} \tilde{A}^c_{ij} \quad (15)$$

Let a value point $(x,u) \in X \times U$, the membership grade $\mu_{\tilde{A}^g}(x,u)$ is computed as follows:

$$\mu_{\tilde{A}^g}(x,u) = f_{ij}(x,u) \quad (16)$$

in which $(x,u) \in D_{ij}$, $i = [(x - x_{min})/dx]$ and $j = [(u - u_{min})/du]$.

Degree of approximation is defined to measure the uncertainty as follows:

Definition 2.4: The degree of approximation (DoA) is the difference between original type-2 fuzzy set $\tilde{A}$ and the representing grid type-2 fuzzy set $\tilde{A}^g$ and is defined as follows:

$$DoA = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \triangle_{ij} / n_c \quad (17)$$

in which $\triangle_{ij} = |\mu_{\tilde{A}}(x^c_i, u^c_j) - f_{ij}(x^c_i, u^c_j)|$, $x^c_i = \frac{1}{2}(\underline{x}_i + \overline{x}_i)$, $u^c_i = \frac{1}{2}(\underline{u}_i + \overline{u}_i)$ and n_c is number of cells that belong to the FOU of $\tilde{A}$.

Let $\tilde{A}^g$, $\tilde{B}^g$ be two grid type-2 fuzzy sets. Note that $\tilde{A}^g$, $\tilde{B}^g$ are discretized on the same grid with dx, du parameters. Results of meet/join operations of $\tilde{A}^g$, $\tilde{B}^g$, called $\tilde{C}^g$, are grid type-2 fuzzy sets with the same partitioned grid that its domain is union of domains $D(\tilde{A}^g)$ and $D(\tilde{B}^g)$. At each vertex (i,j) of the grid of $\tilde{C}^g$, membership grades are computed as follows:

Meet operation:

$$\begin{aligned} \mu_{\tilde{C}}(x_i, u_j) &= \left[max^N_{k=j} \mu_{\tilde{B}^g}(x_i, u_k) \wedge \mu_{\tilde{A}^g}(x_i, u_j)\right] \\ &\vee \left[max^N_{k=j} \mu_{\tilde{A}^g}(x_i, u_k) \wedge \mu_{\tilde{B}^g}(x_i, u_j)\right] \end{aligned} \quad (18)$$

Join operation:

$$\begin{aligned} \mu_{\tilde{C}}(x_i, u_j) &= \left[max^j_{k=0} \mu_{\tilde{B}^g}(x_i, u_k) \wedge \mu_{\tilde{A}^g}(x_i, u_j)\right] \\ &\vee \left[max^j_{k=0} \mu_{\tilde{A}^g}(x_i, u_k) \wedge \mu_{\tilde{B}^g}(x_i, u_j)\right] \end{aligned} \quad (19)$$

in which $\wedge$ is t-conorm operation that may be maximum. $\vee$ is t-norm operation that may be minimum or product.

III. Grid based General Type-2 Fuzzy Logic Systems

A. Inference Engine

This section mentions approach to inference technique for grid general type-2 fuzzy sets with singleton fuzzification that maps the crisp input into a GT2-FS. Singleton fuzzification is considered for which the input GT2FS has only a single point of non-zero membership. Provide that the IF-THEN rules have the form, for l^{th} rule, as follows:

$$R^l : \text{IF } x_1 \text{ is } \tilde{F}_1^l \text{ AND ... AND } x_p \text{ is } \tilde{F}_p^l \text{ THEN } y \text{ is } \tilde{G}^l. \tag{20}$$

where: x_is, y are linguistic variables of inputs, output, respectively; $\tilde{F}_i^l$s are antecedent GT2-FSs (i=1, 2, ..., p); $\tilde{G}^l$ is the output GT2-FS. Provide that all of GT2FS are in the same grid with size $M \times N$.

Inference engine combines rules and gives a mapping from input type-2 fuzzy sets to output type-2 fuzzy sets. Multiple antecedents in rules are connected by the meet operation. The membership grades in the input sets are combined with those in the output sets using the $sup - star$ composition. To do this one needs to find meets and joins of GT2FS, as well as compositions of type-2 fuzzy relations.

Normally, the output of type-2 fuzzy logic systems is crisp value that is defuzzified from output type-2 fuzzy set of inference process. The output of the inference engine is a grid type-2 fuzzy set.

These rules as equation (20) represent a grid based type-2 fuzzy relation between the input space $X_1 \times X_2 \times ... \times X_p$ and the output space Y. The membership function of this type-2 fuzzy relation is denoted as $\mu_{\tilde{F}_1^l \times ... \times \tilde{F}_p^l \to \tilde{G}^l}(x, y)$, where $\tilde{F}_1^l \times ... \times \tilde{F}_p^l$ denotes the Cartesian product of $\tilde{F}_1^l$, ..., $\tilde{F}_p^l$, and $\mathbf{x} = \{x_1, x_2, ..., x_p\}$.

When an input $\mathbf{x}'$ is applied, the composition of the fuzzy set $\tilde{X}'$ to which $\mathbf{x}'$ belongs and the rule R^l is found by using the extended sup-star composition

$$\mu_{\tilde{X}' \circ \tilde{F}_1^l \times ... \times \tilde{F}_p^l \to \tilde{G}^l}(y) = \sqcap_{x \in \tilde{X}'} [\mu_{\tilde{X}'}(x) \sqcap \mu_{\tilde{F}_1^l \times ... \times \tilde{F}_p^l \to \tilde{G}^l}(x, y)] \tag{21}$$

Denote $\tilde{X}' \circ \tilde{F}_1^l \times ... \times \tilde{F}_p^l \to \tilde{G}^l$ as $\tilde{B}^l$ is the output set corresponding to the l^{th} rule. Singleton fuzzification and the product or minimum implication are used, inference process of rule l^{th} is described as follows:

$$\begin{aligned} \mu_{\tilde{B}^l}(y) &= \mu_{\tilde{F}_1^l}(x_1) \sqcap \mu_{\tilde{F}_2^l}(x_2) \sqcap ... \sqcap \mu_{\tilde{F}_p^l}(x_p) \sqcap \mu_{\tilde{G}^l}(y) \\ &= \mu_{\tilde{G}^l}(y) \sqcap [\sqcap_{i=1}^p \mu_{\tilde{F}_i^l}(x_i)] \end{aligned} \tag{22}$$

Denote $f_{\tilde{F}_i^l}(u_j) = \mu_{\tilde{F}_i^l}(x_i)$ where $u_j \in [0, 1]$ and $j = \overline{1, M}$. For any $\tilde{A}^g$, $f_{\tilde{A}^g}(u_j)$ of $\tilde{A}^g$ at x is computed as follows:

$$f_{\tilde{A}^g}(u_j) = (\alpha \mu_{\tilde{A}^g}(x_k, u_j) + \beta \mu_{\tilde{A}^g}(x_{k+1}, u_j) \tag{23}$$

in which $k = (x - x_{min})/dx$, $\alpha = (x - x_k)/dx$ and $\beta = 1 - \alpha$.

For any $\tilde{A}^g$, $\tilde{B}^g$, meet operation of $f_{\tilde{A}^g}(u)$ and $f_{\tilde{B}^g}(u)$ is described as equation (18).

Denote $f^*(u_j) = \sqcap_{i=1}^p f_{\tilde{F}_i^l}(u_j)$. Note that $\tilde{B}^l$ be grid type-2 fuzzy set. Hence, the formula (22) is re-written as follows:

$$\mu_{\tilde{B}^l}(x_i, u_j) = \mu_{\tilde{G}^l}(x_i, u_j) \sqcap f^*(u_j) \tag{24}$$

where $i = \overline{1, N}$, $j = \overline{1, M}$ and $\sqcap$ meet operation is computed as formula (18).

The output sets of rules are combined by using $join$ operation, i.e.

$$\mu_{B^*}(y) = \sqcup_{i=1}^K \mu_{\tilde{B}^i}(y) \tag{25}$$

in which K is number of fired rules and $\mu_{\tilde{B}^i}(y)$ is output of i^{th} fired rule.

Equation (25) can be re-written according to the grid form as follows:

$$\mu_{\tilde{B}^*}(x_i, u_j) = \sqcup_{i=1}^K \mu_{\tilde{B}^i}(x_i, u_j) \tag{26}$$

For each vertical slide at x_i, $\sqcup$ join operation of any $\tilde{A}^g$, $\tilde{B}^g$ is computed as follows:

$$\begin{aligned} f_{\tilde{B}^g \sqcup \tilde{B}^g}(u_j) &= \left[f_{\tilde{A}^g}(u_j) \wedge max_{k=0}^j f_{\tilde{B}^g}(u_k)\right] \\ &\vee \left[f_{\tilde{B}^g}(u_j) \wedge max_{k=0}^j f_{\tilde{A}^g}(u_k)\right] \end{aligned} \tag{27}$$

Before mentioning the algorithm of inference process, the algorithm of meet operation of fuzzified sets are introduced. The algorithms are described as follows:

Algorithm 3.1 (Meet operation of fuzzified sets): .

Input: two fuzzified sets with secondary MFs $f_{\tilde{A}^g}(u)$ and $f_{\tilde{B}^g}(u)$.

Output: output is the secondary $f^*(u)$.

1) Copy data of $f_{\tilde{A}^g}(u)$, $f_{\tilde{B}^g}(u)$, $f^*(u)$ to device memory c_A, c_B, c^*, respectively.
2) Initializing blocks and threads for GPU with number of threads $t_u = 32$ per block. Size of grid: $n_{block} = 1$ and $m_{block} = M/(t_u * k_u)$, where k_u is size of sub-column processed for each thread.
3) For each thread do
   a) Set ix = blockIdx.x*blockDim.x + threadIdx.x;
   b) for $(i = ix * k_x; i < (ix + 1) * k_x; i++)$
      i) Compute $f_1 = min_{k=i}^M c_A[k]$.
      ii) Compute $f_2 = min_{k=i}^M c_B[k]$.
      iii) Compute $c^*[ix] = max(min(f_1, c_B[ix]), min(f_2, c_A[ix]))$.
4) Copy data c^* to host memory $f^*(u)$.

For implementation on GPU platform, the algorithm of inference technique is described as follows:

Algorithm 3.2 (Inference process): .

Input: R GT2-FS based IF-THEN rules as the form (20), p input values x_1, ..., x_p.

Output: GT2-FS $\tilde{B}^*$ is output of inference process.

1) For each rule l^{th}.
   a) Apply each input $x_k, k = \overline{1, p}$, to compute $f_{\tilde{F}_i^l}(u_j)$ as the formula (23).
   b) Compute $f^*(u_j) = f_{\tilde{F}_1^l}(u_j) \sqcap f_{\tilde{F}_2^l}(u_j)$ as the algorithm (3.1).
   c) For k = 3 to p do
      Compute $f^*(u_j) = f^*(u_j) \sqcap f_{\tilde{F}_k^l}(u_j)$ as the algorithm (3.1).
   d) Compute array of max values as $f_{max}(u_i) = max_{k=i}^M f^*(u_k)$.

e) Copy data $\mu_{\tilde{B}^l}(x_i, u_j)$, $f^*(u_j), f^*(u_j)$ to device host m_{B^l}, c^*, c_{max}, respectively.
f) Initializing blocks and threads for GPU with number of threads $t_x = t_u = 32$ per block. Size of grid: $n_{block} = M/(t_x * k_x)$ and $m_{block} = M_{\tilde{C}^g}/(t_u * k_u)$ where k_x, k_u are size of sub-matrix processed for each thread.
g) For each thread do
i) Set ix = blockIdx.x*blockDim.x + threadIdx.x;
ii) Set iu = blockIdx.y*blockDim.y + threadIdx.y;
iii) for $(j = iu * k_u; j < (iu + 1) * k_u; j++)$
A) Compute $f_1 = max_{k=j}^{M} m_{B^l}[k, j]$.
B) for $(i = ix * k_x; i < (ix + 1) * k_x; i++)$
C) Compute $m_{B^l}[i, j] = max(min(f_1, c^*[j]), min(m_{B^l}[i, j], c_{max}[j]))$.
h) Copy data of result from device to host memory.
2) If $p = 1$ then return $\mu_{\tilde{B}^1}(x_i, u_j)$
3) Else
a) Compute $\mu_{\tilde{B}^*}(x_i, u_j) = \mu_{\tilde{B}^1}(x_i, u_j) \sqcup \mu_{\tilde{B}^2}(x_i, u_j)$ as the formula (19).
b) For k = 3 to K do
c) Compute $\mu_{\tilde{B}^*}(x_i, u_j) = \mu_{\tilde{B}^*}(x_i, u_j) \sqcup \mu_{\tilde{B}^k}(x_i, u_j)$ as the formula (19).
4) return $\mu_{\tilde{B}^*}(x_i, u_j)$.

B. Defuzzification

This operation is to find the centroid of uncertainty of space of grid type-2 fuzzy sets. So the section mentions an algorithm to compute the 3-dimensional centroid of the grid by combining from centroids of cells. The following is outline of defuzzification algorithm:

Algorithm 3.3: Defuzzification operation
Input: $\tilde{A}^g$ is a grid type-2 fuzzy sets.
Output: value of defuzzification.

1) Initializing s_x, s_u are sizes of sub-matrix for processing of each thread.
2) Initializing device memory m_A for matrices of grid $\tilde{A}^g$.
3) Copy data from host memory to device memory of grid $\tilde{A}^g$.
4) Initializing blocks and threads for GPU. We implement with number of threads $t_x = t_u = 32$ per block. So size of grid is computed as $n_{block} = M_{\tilde{C}^g}/(t_x * k_x)$ and $m_{block} = M_{\tilde{C}^g}/(t_u * k_u)$ where k_x, k_u are size of sub-matrix processed for each thread.
5) Initializing device memory m_d, m_u for storing data for each thread.
6) Call ds be area of a cell in the grid of T2FS.
7) $f1 = f2 = 0$
8) For each thread do
a) Set ix = blockIdx.x*blockDim.x + threadIdx.x;
b) Set iu = blockIdx.y*blockDim.y + threadIdx.y;
c) for $(i = ix * k_x; i < (ix + 1) * k_x; i++)$
d) for $(j = iu * k_u; j < (iu + 1) * k_u; j++)$
i) $fa = (m_A[i, j] + m_A[i + 1, j] + m_A[i, j + 1] + m_A[i + 1, j + 1])/4.0$.
ii) $fu = (j + 0.5)/N$.
iii) $f1 += fa * ds * (i + 0.5) * fu$.
iv) $f2 += fa * ds * fu$.
e) $m_d[ix, iu] = f1$, $m_u[ix, iu] = f2$.
9) $f1 = f2 = 0$.
10) Copy data from device memory of m_d, m_u to host memory.
11) Compute sum of $m_d[i, j]$s, call s_1, and sum of $m_u[i, j]$s, call s_2.
12) Value of defuzzification is s_1/s_2.

Hence, the algorithm is implemented on both of GPU (at threads) and CPU, combination of results of threads.

IV. Experiments

In this section, we mention experiments on representation and operations of grid type-2 fuzzy sets involving join , meet, negation operation and defuzzification. All experiments are implemented on ASUS Laptop with CPU CORE^{TM} i5, 6GB RAM, Windows 7 64bit and nVIDIA GeForce GT520M using VC 2008 Release Compiler and CUDA SDK 5.0.

Let $\tilde{A}$ is a general type-2 fuzzy set $g_{\tilde{A}}(m_1, m_2, \sigma)$. The feature membership functions of $\tilde{A}$ are described as follows:

FOU is Gaussian function with upper MF and lower MF as follows:

Upper MF of FOU:

$$f_u(x) = \begin{cases} e^{-\frac{1}{2}(\frac{x-m_1}{\sigma})^2} & if\ x < m_1 \\ 1 & if\ m_1 \le x \le m_2 \\ e^{-\frac{1}{2}(\frac{x-m_2}{\sigma})^2} & if\ x > m_2 \end{cases} \quad (28)$$

Lower MF of FOU:

$$f_l(x) = \begin{cases} e^{-\frac{1}{2}(\frac{x-m_2}{\sigma})^2} & if\ x < \frac{m_1+m_2}{2} \\ e^{-\frac{1}{2}(\frac{x-m_1}{\sigma})^2} & if\ otherwise \end{cases} \quad (29)$$

where $m_1 = 3.0$, $m_2 = 4.0$ and $\sigma = 0.5$.

The next feature of $\tilde{A}$ is set of points where $\mu_{\tilde{A}}(x, u) = 1.0$, involves points belong to the MF described as follows:

$$f_m(x) = e^{-\frac{1}{2}(\frac{x-(m_1+m_2)/2}{\sigma})^2} \quad (30)$$

The secondary membership function at x of $\tilde{A}$ are Gaussian functions that are described as follows:

$$g(u) = \begin{cases} e^{-\frac{1}{2}(\frac{u}{\sigma_1})^2} & if\ u \ge u_0 \\ e^{-\frac{1}{2}(\frac{u}{\sigma_2})^2} & if\ u < u_0 \end{cases} \quad (31)$$

in which $u_0 = f_m(x)$, $\sigma_1 = 3.035 * |\overline{u} - u_0|$ and $\sigma_2 = 3.035 * |\underline{u} - u_0|$.

A. Defuzzification

Defuzzification operation is implemented on $\tilde{A}^g$, the Gaussian $g_{\tilde{A}}(3, 4, 0.6)$. The precise value of defuzzification of $\tilde{A}^g$ is 3.5. Difference between the precise value and the output of defuzzification is enough small in various resolution. High resolution of grid results in high accuracy of defuzzification.

TABLE I
RUNTIME AND DIFFERENCE VALUE OF DEFUZZIFICATION OPERATION

Difference	128×32	512×128	2048×512	8192×1024
CPU (ms)	0.067	1.075	48.161	434.821
Resolution	5.5e-5	5.0e-6	4.0e-5	9.83e-4
GPU (ms)	0.75	0.755	5.202	39.272
Difference	5.81e-4	4.0e-6	7.0e-6	1.7e-5
CPU/GPU	0.088	1.433	9.259	11.072

TABLE II
RUNTIME OF INFERENCE TECHNIQUE WITH VARIOUS GRIDS (IN MILLISECONDS)

	$2^7 \times 2^5$	$2^9 \times 2^7$	$2^{11} \times 2^9$	$2^{13} \times 2^{10}$
CPU	0.90	51.35	3671.61	57433.14
GPU	1.43	5.68	168.61	2467.18
CPU/GPU	0.63	9.04	21.78	23.27

B. Inference Technique

For implementation of inference process, a rule base involving two rules is described as follows:

IF x_1 is $g_{\tilde{A}_1}(3, 4, 0.5)$ AND x_2 is $g_{\tilde{B}_1}(4, 4.5, 0.2)$ THEN y is $g_{\tilde{B}_1}(3, 3.5, 0.3)$.

IF x_1 is $g_{\tilde{A}_2}(4.5, 5.5, 0.4)$ AND x_2 is $g_{\tilde{B}_2}(4.5, 5.5, 0.2)$ THEN y is $g_{\tilde{B}_1}(4.5, 5.5, 0.3)$.

Note that $g_{\tilde{A}}(m_1, m_2, \sigma)$ be above described Gaussian grid general type-2 fuzzy sets. Fig. 3(a) depicts GT2FSs of above rules. The inference process is implemented by applying two input values $x_1 = 4$, $x_2 = 4.5$ to grid T2FS of antecedent of rules. Fig. 3(b) describes output of rules from implication and the final grid T2FS after combining output of rules using join operation. Table II shows summarised of implementation on CPU and GPU platforms. Algorithms are implemented 10 times for taking average of run-times. As summarised report in Table II, when size of grid is large, algorithms of CPU platform take huge run-times meanwhile run-times of GPU algorithms are allowable to deploy real applications with high accuracy. The run-time of inference process depends on number of inputs p and rules R. The number of operations involves $R \times (p-1)$ meet operations between column type-2 FSs, R meet operations between column and grid T2FS, $R-1$ join operations between grid T2FSs.

C. Robot navigation

We implement type-2 fuzzy logic systems with collision avoidance behavior of robot navigation. The fuzzy logic systems have two inputs: the extended fuzzy directional relation [19] and range to obstacle, the output is angle of deviation (AoD). The fuzzy rule has the form as following:

IF FDR is $\tilde{A}_i$ *AND Range is* $\tilde{B}_i$ *THEN AoD is* $\tilde{C}_i$

where $\tilde{A}_i$, $\tilde{B}_i$, $\tilde{C}_i$ are type-2 fuzzy sets of antecedent and consequent, respectively.

The fuzzy directional relation has six linguistic values (NLarge, NMedium, NSmall, PSmall, PMedium and PLarge). The range from robot to obstacle is divided in three subsets: VNear, Near, Medium and Far. The output of fuzzy if-then is a linguistic variable representing for angle of deviation, has six linguistic variables the same the fuzzy directional relation with the different membership functions. Linguistic values are general type-2 fuzzy subsets that membership functions are described in Fig. 4. Membership functions are Gaussian GT2FS mentioned above. The rule-base is described in the table III.

TABLE III
THE RULE BASE OF COLLISION AVOIDANCE BEHAVIOR

FDR	Range	AoD	FDR	Range	AoD
NS	N	PL	PS	N	NL
NS	M	PM	PS	M	NM
NS	F	PS	PS	F	NS
NM	N	PM	PM	N	NM
NM	M	PM	PM	M	NM
NM	F	PS	PM	F	NS
NL	N	PM	PL	N	NM
NL	M	PS	PL	M	NS
NL	F	PS	PL	F	NS

General type-2 FLS is tested on the whole of discretized input spaces with step of 0.1 in which $FDR \in [-1, 1]$ and $Range \in [0.0, 2.0]$, i.e we have the whole of 200×200 input points. Table IV shows the run-time of inference process for 400 input points and the result of inference process is shown in Fig. 5.

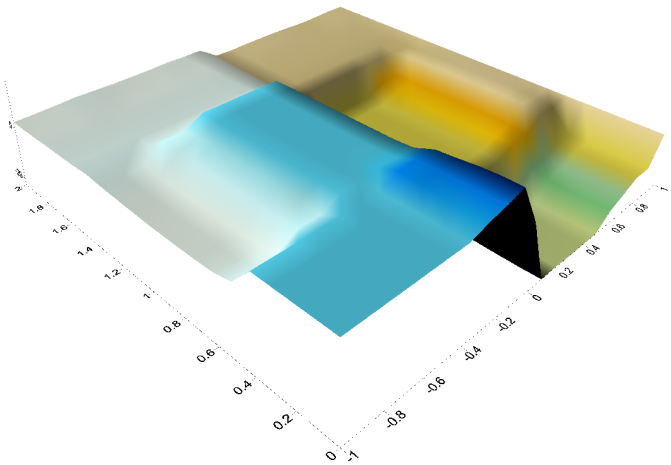

Fig. 5. Inference Surface of Type-2 avoidance behavior

TABLE IV
RUNTIME OF INFERENCE PROCESS OF THE WHOLE OF DISCRETIZED INPUT SPACE(IN SECONDS)

	512×128	1024×256	2048×512	4096×1024
CPU	12.33	85.28	658.40	4368.01
GPU	2.13	5.61	23.50	107.00
CPU/GPU	5.79	15.2	28.0	40.8

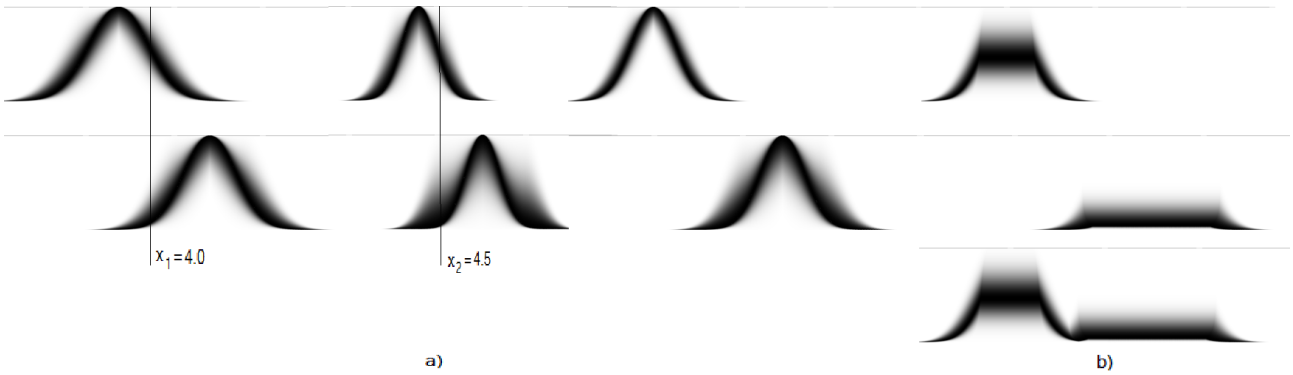

Fig. 3. Explanation of inference process for grid T2FS based rules.

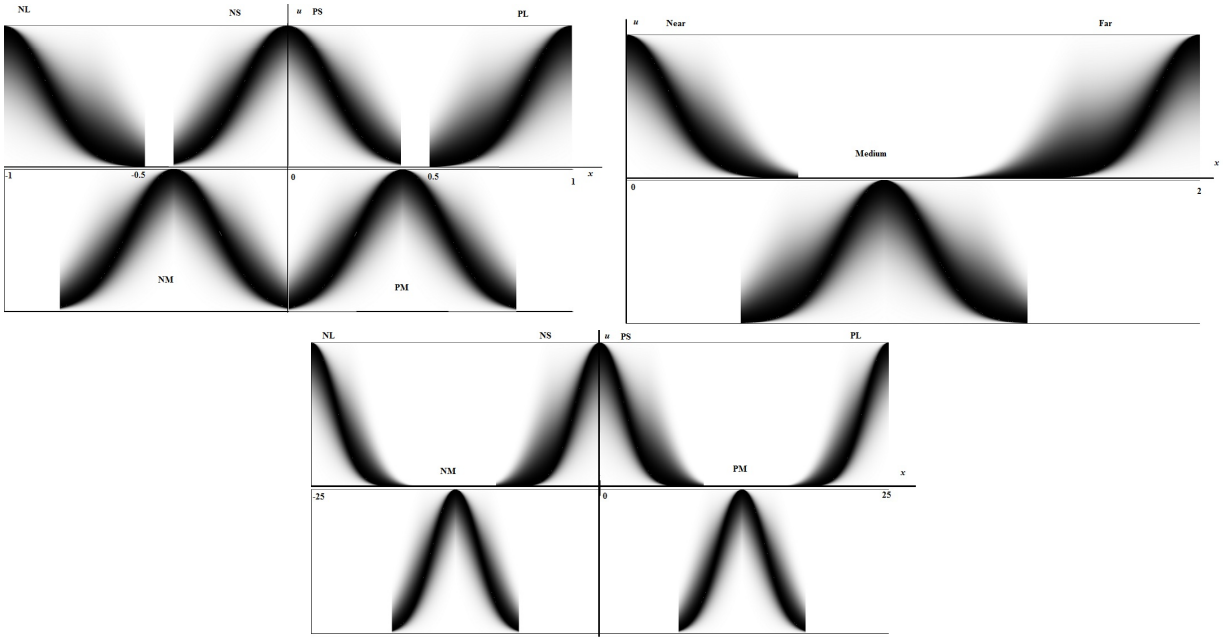

Fig. 4. Membership grades of CA behavior.

V. Conclusion

The paper has presented an approach to representation of grid general type-2 fuzzy sets for speed-up of computation based on GPU. The inference process of general type-2 FLS is proposed with implementation. The approach is to speed up computation of general T2FLS in the case of high accuracy of GT2FS. The result points out that high performance is even obtained at low hardware of nVIDIA card. In the case of higher nVIDIA hardware with multi cores, the high performance is many times higher than on CPU platform.

References

[1] KC Zikidis, "ASAFES2: A novel neuro-fuzzy architecture for fuzzy computing, based on functional reasoning", *Fuzzy Sets and Systems*, Vol. 83, no.1, pp.63-84, 1995.

[2] Witold Pedrycz, Vasilakos, A.V., "Linguistic models and linguistic modeling", *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, Vol. 29, no.6, pp. 745-757, 1999.

[3] N. Karnik, J.M. Mendel, Liang Q. "Type-2 Fuzzy Logic Systems", *IEEE Trans. on Fuzzy Systems*, Vol.7, No. 6, pp. 643-658, 1999.

[4] N. Karnik, J.M. Mendel, "Operations on Type-2 Fuzzy Sets", *Fuzzy Sets and Systems*, Vol. 122, pp.327-348, 2001.

[5] J.M. Mendel, R. I. John, "Type-2 Fuzzy Sets Made Simple", *IEEE Trans. on Fuzzy Systems*, Vol.10, no. 2, pp. 117-127, 2002.

[6] L. T. Ngo, L. T. Pham, P. H. Nguyen, "Extending fuzzy directional relationship and applying for mobile robot collision avoidance behavior", *Int' Journal of Advanced Computational Intelligence & Intelligent Informatics*, Vol. 10, no.4, pp.444–450, 2006.

[7] S.Coupland, R.John, "Geometric Type-1 and Type-2 Fuzzy Logic System", *IEEE Trans. on Fuzzy Systems*, Vol. 15, no. 1, pp. 3-15, 2007.

[8] X. Ban, Gao, X.Z.; Xianlin Huang; Hang Yin, "Stability analysis of the simplest Takagi-Sugeno fuzzy control system using circle criterion", *Information sciences*, vol. 177, no. 20, pp.4387-4409, 2007.

[9] S.Coupland, R.John, "New geometric inference techniques for type-2 fuzzy sets", *Int' Journal of Approximate Reasoning*, Vol. 49, no.1, pp. 198-211, 2008.

[10] S. Coupland, R. John, "A Fast Geometric Method for Defuzzification of Type-2 Fuzzy Sets", *IEEE Trans. on Fuzzy Systems*, Vol. 16, no.4, pp. 929-941, 2008.

[11] F. Liu, "An efficient centroid type-reduction strategy for general type-2 fuzzy logic system", *Information Sciences*, vol.178, no.9, pp. 2224-2236, 2008.

[12] D.Anderson, R.Luke, J.Keller, "Speed-up of Fuzzy Clustering Through Stream Processing on Graphics Processing Units", *IEEE Trans. on Fuzzy Systems*, Vol. 16, no.4, pp. 1101-1106, 2008.

[13] Anderson, "Parallelisation of Fuzzy Inference on a Graphics Processor Unit Using the Compute Unified Device Architecture", in *The 2008 UK Workshop on Computational Intelligence*, pp. 1-6, 2008.

[14] N.Harvey, R.Luke, J. Keller, D.Anderson, "Speed-up of Fuzzy Logic through Stream Processing on Graphics Processing Units", in *Proc. IEEE Congress on Evolutionary Computation*, pp.3809-3815, 2008.

[15] J. M. Mendel, F. Liu, D. Zhai, "α-Plane Representation for Type-2 Fuzzy Sets: Theory and Applications", *IEEE Trans. on Fuzzy Systems*, Vol.17, no.5, pp. 1189-1207, 2009.

[16] Janusz T. Starczewski, "Efficient triangular type-2 fuzzy logic systems", *International Journal of Approximate Reasoning*, Vol. 12, no.5, pp. 799-811, 2009.

[17] C. Wagner, H. Hagras, "Toward General Type-2 Fuzzy Logic Systems Based on zSlices", *IEEE Trans. on Fuzzy Systems*, Vol. 18, no.4, pp. 637-660, 2010.

[18] K.Sejun, C.Donald, "A GPU based Parallel Hierarchical Fuzzy ART Clustering", in *Proceedings of International Joint Conference on Neural Networks*, vol.1, pp.2778-2782, 2011.

[19] F.Chia, C.Teng, Y.Wei, "Speed-up of Implementing Fuzzy Neural Networks With High-Dimensional Inputs Through Parallel Processing on Graphic Processing Units", *IEEE Trans. on Fuzzy Systems*, Vol.19, no.4, pp.717-728, 2011.

[20] D.Zhai, J.M.Mendel, "Computing the Centroid of a General Type-2 Fuzzy Set by Means of the Centroid-Flow Algorithm", *IEEE Trans on Fuzzy Systems*, Vol. 19, no.3, pp. 401–422, 2011.

[21] O. Linda, M. Manic, "Monotone Centroid Flow Algorithm for Type-Reduction of General Type-2 Fuzzy Sets", *IEEE Trans on Fuzzy Systems*, Vol.20, no.5, pp. 805 - 819, 2012.

[22] Long Thanh Ngo, "Operations of grid general type-2 fuzzy sets based on GPU computing platform", in *Proceedings of IEEE SMC 2012*, pp. 2885-2890, 2012.

[23] L. T. Ngo, "General type-2 fuzzy logic systems based on refinement constraint triangulated irregular network", *Journal of Intelligent and Fuzzy Systems*, 2013, in press.