Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Novel Method for Determining the Centroid of Type-3 Fuzzy Sets

Jagannath Biswas
University of Calcutta
Samarjit Kar
samarjit.kar@maths.nitdgp.ac.in
NIT Durgapur
Tripti Chakrabarti
University of Calcutta

Research Article

Keywords: Type-2 Fuzzy Sets, Type-3 Fuzzy Sets, Interval Type-3 Fuzzy sets, Centroid

Posted Date: March 11th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-8883447/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

A Novel Method for Determining the Centroid of Type-3 Fuzzy Sets

Jagannath Biswas[1], Samarjit Kar[2,3*], Tripti Chakrabarti[4,5]

[1]Department of mathematics, Lalbaba College, University of Calcutta, Kolkata, India
[2]Department of Mathematics, NIT Durgapur, India
[3]Department of Graphical Systems, Vilnius Gediminas Technical University, Lithuania
[4]Department of mathematics, University of Calcutta, Kolkata, India
[5]Department of Basic Science, Techno India University, Kolkata, India

Abstract: Finding the centroid of a fuzzy set is one of the most crucial and significant computational tasks in fuzzy logic systems. Since Zadeh's time, centroids of type-1 and type-2 fuzzy sets have only been used in type-reduction. Although there are some applications for type-3 fuzzy control, there is currently no literature on the usage of centroid during type-reduction. The existing literature currently lacks a method for determining the centroid of the type-3 fuzzy set. In this study, we have attempted a relatively straightforward but rigorous way to find the centroid of a type-3 fuzzy set using weights in each direction of membership variables. This method may also be used to type-2 and type-1 fuzzy sets. Examples are provided that compare the computational results and show the computational efficiency of the proposed method.

Keywords: Type-2 Fuzzy Sets; Type-3 Fuzzy Sets; Interval Type-3 Fuzzy sets; Centroid

1. Introduction

A fuzzy logic system (FLS) comprised of three steps in case of type-1 fuzzy input and/or rule and five steps in case of type-2 or higher. A FLS is referred to as type-1 FLS when it receives a type-1 fuzzy set as both an input and an inference rule, and type-2 FLS if any one of these elements is a type-2 fuzzy set.

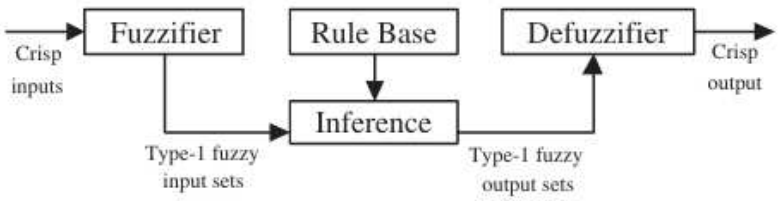


Fig 1: Type-1 Fuzzy Logic System

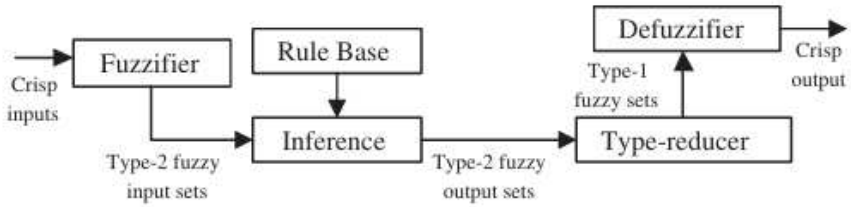


Fig 2: Type-2 Fuzzy Logic System

When the input/rules are type-1, then output in the FLS is also type-1 and only defuzzification is enough to find crisp output as depicted in figure 1. But when it is type-2 FLS, before defuzzification it is necessary to find a type-1 fuzzy set by type-reduction first. Therefore, type-2 FLS requires an additional step, as illustrated in figure 2. Several research articles about FLS have been published up to date. In this study, we attempt to concentrate on articles that are directly related to the centroid of a fuzzy set.

Fuzzy set theory originated by Zadeh [31] in his seminal paper of 1965. After that when Mamdani started work on fuzzy logic system from 1970s, centroid was used as a defuzzification technique. Later on, in 2001, Karnik and Mendel [1] developed an iterative method (popularly known as "KM-algorithm") for finding centroid of interval type-2 fuzzy set (IT2FS). There has been a noticeable increase in the rate of publication for this topic since this paper. In 2002, Wu and Mendel [20] found inner and outer bound of the type-reduced set of fuzzy logic system on the basis of KM-algorithm. In 2006, Wu and Mendel [21] explained centroid as measure of uncertainty of type-2 fuzzy set. Then, in 2008, Liu and Mendel [4] did an excellent job by deriving the analytical convergence of KM algorithm. In 2009, Wu and Mendel [2] did some work on the previous paper of KM algorithm by introducing a better initialization through simulation results, which became also popular by the name "Enhanced KM (EKM) algorithm". An opposite direction searching (ODS) algorithm was proposed by Hu et al. [22] for computing the centroid of IT2FS. Later, Hu et al. [23] improved the performance of the ODS algorithm with two high speed formulae for computing the centroid endings, known as Enhanced ODS (EODS) algorithm. Duran et al. [3] proposed an iterative process with stopping condition, popularly known as IASC algorithm. Then Wu and Maowen [26] better developed the enhanced part of IASC, known as EIASC. In 2012, Melgarejo and Celemin [24] proposed another iterative process. For general type-2 fuzzy set (GT2FS), in 2010 Zhai and Mendel [8] developed centroid flow algorithm, which used alpha-plane representation of GT2FS. In 2011, they again did some improvement by changing the starting procedure, known as ECF algorithm [16]. Besides these papers, some non-iterative procedures also proposed in the literature. Nie-Tan [27] found a non-iterative closed formula for IT2FS; Begian et al. [28], [29], [30] did some work on robustness, stability of type-2 fuzzy system.

But after 2012, we did not notice any such theoretical development on the way of higher type fuzzy set (either in interval type or general type) centroid, only some review paper and hundreds of applications in different hybrid intelligence techniques have been published. Although after 2022, Castilo et al. [17] and some other researchers have published some papers on type-3 fuzzy set but most of them on applications. Recently, Mohammadzadeh et al. [14] published a paper about uncertainty bound of centroid of interval type-3 fuzzy set, which can be resembled to the paper of Wu and Mendel [20], except the introduction of a new optimization algorithm.

The main findings of this study are as follows:

- We attempted to begin investigating the method of determining the centroid of a fuzzy set for any type, whether it be type-1, type-2, type-3, or type-n, and whether it is interval type or general type.
- We have also highlighted some shortcomings of previous algorithms, as pointed out in the motivation part of section 4.

The organization of this study is as follows. Background information on fuzzy sets is presented in Section 2. Section 3 displays some existing algorithms in detail. Our proposed technique is developed in section 4. Section 5 shows the computational complexity of our method. We have compared our method with other methods in section 6. The main conclusions and future directions are summarized in Section 7.

2. Preliminaries

In this section, some notations and definitions concerning different type of fuzzy sets are introduced. For visualisation, see figure 3.

Fuzzy Set: $A = \{(x, \mu_x): x \in X, 0 \leq \mu_x \leq 1\}$ is a type-1 fuzzy set (T1FS). Here, μ_x is the membership value of the point x in the universe of discourse X.

Type 2 fuzzy set: $\tilde{A} = \{(x, u, \mu_u): x \in X, u \in J_x \subseteq [0,1],\ \mu_u: J_x \to [0,1]\}$ is a type-2 FS (T2FS). Here, membership value of a point x in X, is not a crisp point like T1FS, rather it is a T1FS itself, whose universe of discourse is $J_x \subseteq [0,1]$ for a point x in X. and the secondary membership is represented by the function $\mu_u: J_x \to [0,1]$.

- If $\mu_u(u_i) = 1, \forall\ u_i \in J_x$, then $\tilde{A}$ is an ***Interval Type-2 Fuzzy Set (IT2FS)***
- If the secondary membership value $\mu_u(u_i)$ is different for all $u_i \in J_x$, then $\tilde{A}$ is called ***General Type-2 Fuzzy Set (GT2FS)***.

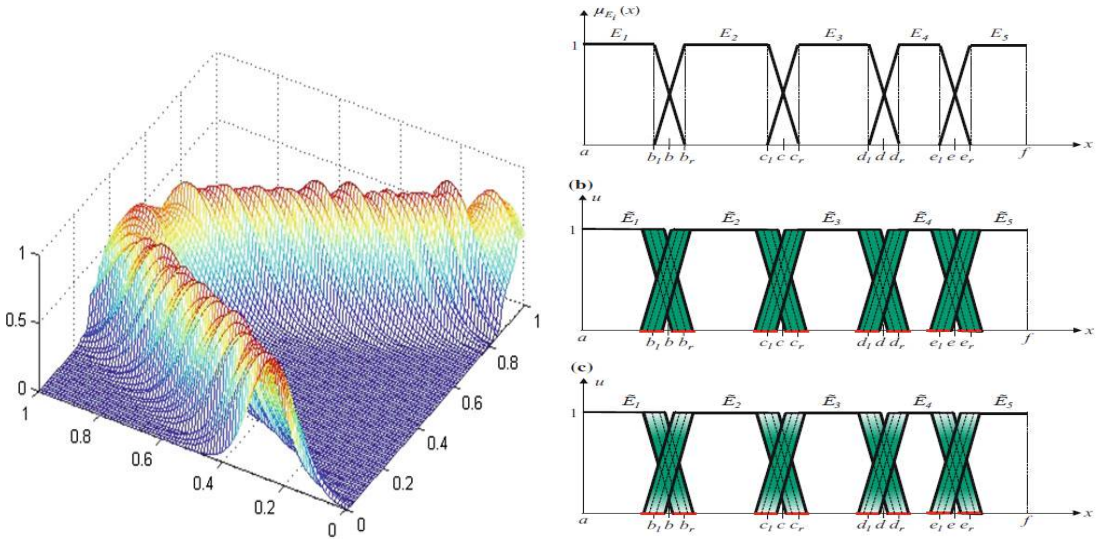


Fig 3: Left side is a 3D presentation of a GT2FS. Right side presents T1FS, IT2FS and GT2FS respectively. Pictures taken from [19]

Some details about IT2FS:

- FOU $(\tilde{A}) = \cup_{x \in X} J_x = [\underline{A}(x), \overline{A}(x)]$, where $\underline{A}(x)$ is LMF and $\overline{A}(x)$ is UMF for an IT2FS.
- $\tilde{A} = 1/\text{ FOU }(\tilde{A})$.
- $A_e = \sum_{i=1}^{N} {}^{u_i}/_{x_i}$, where $u_i \in J_{x_i} \subseteq U = [0,1]$.
- Representation Theorem for IT2FS: - FOU $(\tilde{A}) = \cup_{i=1}^{n_A} A_e^j = \{\underline{A}(x), \dots, \overline{A}(x)\} = [\underline{A}(x), \overline{A}(x)]$, where $n_A = \prod_{i=1}^{N} M_i$.

Type-3 fuzzy set: $\tilde{A} = \{(x, u, w, \mu_w): x \in X, u \in J_x \subseteq [0,1], w \in k_u \subseteq [0,1], \mu_w: k_u \to [0,1]\}$ is a type-3 FS (T3FS). Here, membership value of a point x in X, is not a T1FS, rather it is a T2FS itself, whose universe of discourse is $J_x \subseteq [0,1]$ for a point x in X. and the secondary membership is represented by the function μ_u: $J_x \to [0,1]$ and the tertiary membership function is represented by μ_w: $k_u \to [0,1]$, where k_u is the set of all possible membership values of u. We can understand it later geometrically.

- If $\mu_w(w_i) = 1, \ \forall\, w_i \in k_u$, then $\tilde{A}$ is an ***Interval Type-3 Fuzzy Set (IT2FS).*** Geometrically we can visualise as x in dimension 1, u in dimension 2, w in dimension 3, see figures 10 and 11.

- If the secondary membership value $\mu_w(w_i)$ is different for all $w_i \in k_u$, then $\tilde{A}$ is called ***General Type-3 Fuzzy Set (GT3FS).*** Geometrically it is a bit tough to visualise GT3FS, although we have tried to represent it at the end of section 4.

3. Existing Techniques for Finding Centroid

Several studies on this topic have been published thus far. Just a few of them are novel techniques. Some techniques are more promising and those are highlighted in the literature of type-reduction part. We can classify these methods in two categories. The first one is for interval type-2 fuzzy set (IT2FS) and the second one for general type-2 fuzzy set (GT2FS).

The Karnik-Mendel (KM), Enhanced KM (EKM), Iterative Algorithm with Stopping Condition (IASC), and Enhanced IASC (EIASC) algorithms are the most significant iterative techniques for IT2FS. The only non-iterative techniques are the Nie-Tan (NT), Nagar-Bardini (NB), and Begian-Melek-Mendel (BMM) methods. Each iterative process is predicated on the method produced by the KM algorithm. Therefore, we first highlight the processes in the KM algorithm, and then discuss each change or modification for the other techniques.

KM- algorithm [1]: In an IT2FS $\tilde{A}$, the centroid is a T1FS $[c_l(L), c_r(R)]$ whose midpoint is considered as defuzzified crisp value of the output of a FLS. The formulas to find c_l, c_r are as follows:

Step 1:

$$c_l(L) = \min_{\forall\theta_i \in \left[\underline{\mu}_{\tilde{A}}(x_i), \overline{\mu}_{\tilde{A}}(x_i)\right]} \left(\sum_{i=1}^{N} x_i\theta_i \Big/ \sum_{i=1}^{N} \theta_i\right),$$

$$c_r(R) = \max_{\forall\theta_i \in \left[\underline{\mu}_{\tilde{A}}(x_i), \overline{\mu}_{\tilde{A}}(x_i)\right]} \left(\sum_{i=1}^{N} x_i\theta_i \Big/ \sum_{i=1}^{N} \theta_i\right).$$

To find these: initialize θ_i by setting $\theta_i = \left(\underline{\mu}_{\tilde{A}}(x_i) + \overline{\mu}_{\tilde{A}}(x_i)\right)/2,\ i = 1, 2, \ldots, N.$ and then compute

$$c' = \sum_{i=1}^{N} x_i\theta_i \Big/ \sum_{i=1}^{N} \theta_i$$

Step 2: find k such that $x_k \le c' \le x_{k+1}$

Step 3: set $\theta_i = \overline{\mu}_{\tilde{A}}(x_i)$ when $i \le k$, and $\theta_i = \underline{\mu}_{\tilde{A}}(x_i)$ when $i \ge k+1$ and then compute

$$c_l(k) = \frac{\sum_{i=1}^{k} x_i\overline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} x_i\underline{\mu}_{\tilde{A}}(x_i)}{\sum_{i=1}^{k} \overline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} \underline{\mu}_{\tilde{A}}(x_i)}$$

Step 4: if $c_l(k) = c'$, then STOP and assign $k = L$. if no, go to step 5

Step 5: set $c' = c_l(k)$ and go to step 2.

A similar technique for $c_r(R)$.

EKM algorithm [5]: Here, three approaches are different, such as (i) a better initialization is used to reduce the number of iterations; (ii) the termination condition of the iterations is changed to remove an unnecessary iteration; and (iii) a subtle computing technique is used to reduce the computational cost of each of the algorithm's iterations

Step 1: $k = [N/2.4]$, $c^{/} = \frac{a}{b} = \frac{\sum_{i=1}^{k} x_i \overline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} x_i \underline{\mu}_{\tilde{A}}(x_i)}{\sum_{i=1}^{k} \overline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} \underline{\mu}_{\tilde{A}}(x_i)}$

Step 2: find $k^{/}$ such that $x_{k^{/}} \leq c^{/} \leq x_{k^{/}+1}$

Step 3: if $k^{/} = k$, then STOP and set $c^{/} = c_l(L)$ and $k = L$. if not, go to step 4

Step 4: compute $s = \text{sign}\,(k^{/} - k)$, and compute

$$c^{//}(k^{/}) = \frac{a^{/}}{b^{/}} = \frac{a + s\sum_{i=\min\left(k^{/},k\right)+1}^{\max\,(k^{/},k)} x_i\left(\overline{\mu}_{\tilde{A}}(x_i) - \underline{\mu}_{\tilde{A}}(x_i)\right)}{b + s\sum_{i=\min\left(k^{/},k\right)+1}^{\max\,(k^{/},k)} \left(\overline{\mu}_{\tilde{A}}(x_i) - \underline{\mu}_{\tilde{A}}(x_i)\right)}$$

Step 5: set $c^{/} = c^{//}(k^{/})$, $a = a^{/}, b = b^{/}$ and go to step 2.

Same applied for right end point $c_r(R)$ of the centroid interval.

EIASC Algorithm: Mendel et al. [5] stated that any optimization algorithm *requires* a good way to (i) initialize it, (ii) move from one step to the next, and (iii) stop. All of the algorithms devised for this topic focus on at least one of these criteria. A *brute-force* way to find $c_l(L)$ or $c_r(R)$ is to compute $\{c_l(k)\}_{k=1}^{N-1}$ or $\{c_r(k)\}_{k=1}^{N-1}$ and then locate its smallest (largest) element. This is called "brute force" (or exhaustive) because it does not focus on a good way to initialize the process, move from one step to the next, or stop. Malgarejo [26] and Duran et al. [3] tried to approach by the brute force algorithm in iterative algorithm plus stopping condition (IASC) and after that Wu and Nie [27] did some improvements as it was popular as EIASC. Table 0 illustrates the corresponding algorithm.

Table 0: EIASC algorithm-results taken from Mendel et al. [5]

Step	EIASC for $c_l(L)$	EIASC for $c_r(R)$
1	Initialize	Initialize
	$a = \sum_{i=1}^{N} x_i \underline{\mu}_{\tilde{A}}(x_i)$	$a = \sum_{i=1}^{N} x_i \overline{\mu}_{\tilde{A}}(x_i)$
	$b = \sum_{i=1}^{N} \underline{\mu}_{\tilde{A}}(x_i)$	$b = \sum_{i=1}^{N} \overline{\mu}_{\tilde{A}}(x_i)$
	$L = 0$	$R = N$
2	Compute	Compute
	$L = L + 1$	$a = a + x_R[\overline{\mu}_{\tilde{A}}(x_R) - \underline{\mu}_{\tilde{A}}(x_R)]$
	$a = a + x_L[\overline{\mu}_{\tilde{A}}(x_L) - \underline{\mu}_{\tilde{A}}(x_L)]$	$b = b + [\overline{\mu}_{\tilde{A}}(x_R) - \underline{\mu}_{\tilde{A}}(x_R)]$
	$b = b + [\overline{\mu}_{\tilde{A}}(x_L) - \underline{\mu}_{\tilde{A}}(x_L)]$	$c_r(R) = a/b$
	$c_l(L) = a/b$	$R = R - 1$
3	If $c_l(L) \le x_{L+1}$, stop, otherwise go to step 2	If $c_r(R) \ge x_R$, stop, otherwise go to step 2

In addition to these methods, iterative process for general type-2 fuzzy sets is being popular in practice. In last 15 years almost, only two methods other than KM and EKM algorithms are Enhanced Central Flow (ECF) algorithm and Improved Enhanced Opposite Direction Searching (IEODS) algorithm. It should be remembered that KM and EKM algorithms not only useful for IT2FS but also for GT2FS as the main algorithm for finding the centroid of a general type-2 fuzzy set (GT2FS) is to first decompose it into a series of alpha-planes and then compute the centroid of each plane using an iterative algorithm like the KM or EKM.

The Centroid Flow Algorithm evaluates the centroids of two end α-planes first using the KM or EKM algorithm, then letting the results flow to the centroids of all other α-planes using specific connecting equations.

Before starting the algorithms lets know some background of GT2FS.

A GT2FS $\tilde{A}$ can be written in many ways. In vertical slice representation [9] it can be written as $\tilde{A} = \int_{\forall x \in X} \left[\int_{\forall u \in J_x} \mu_x(u)/u \right] / x$, where x, u are the primary and secondary variable respectively, $J_x (\subseteq [0,1])$is the primary membership of x and $\mu_x(u)$ is the secondary membership (which can be regarded as primary membership of u). But in wavy-slice representation [6], $\tilde{A}$ can be written as union of all embedded T2FS $\widetilde{A_e}$, where $\widetilde{A_e} = \int_{x \in X} [\mu_x(u)/u]/x$ (here, every x has a single membership u), i.e., $\tilde{A} = \bigcup_{\forall \widetilde{A_e}} \widetilde{A_e}$.

Another important thing is FOU($\tilde{A}$) which is the shaded area of the figure(a), which can be expressed as

$$\int_{\forall x \in X} \left[\int_{\forall u \in J_x} (\mu_x(u) \ge 0)/u \right] / x.$$

$\boldsymbol{\alpha - plane\ representation}$ [7] is also a major pillar in the development of the literature involving centroid of GT2FS. $\alpha - plane$ of a GT2FS is nothing but the analogy of $\alpha - cut$ in a T1FS, i.e., the union of all primary memberships whose secondary memberships are greater than or equal to α, so that $\alpha - plane\ \widetilde{A_\alpha} = \int_{\forall x \in X} \left[\int_{\forall u \in J_x} (\mu_x(u) \geq \alpha)/u\right]/x = \cup_{\forall A_e(\alpha)} A_e(\alpha)$, where $A_e(\alpha)$ is an embedded T1FS of the embedded T2FS $\widetilde{A_e}$, lying inside FOU($\tilde{A}$). Also, a very interesting aspect here we observe that, if we vary α, we will get a nested α −plane on the $x - u$ plane. For $\alpha = 1$, the α −plane is a single embedded T1FS $A_e(1)$ as a curve lied on $x - u$ plane inside the FOU($\tilde{A}$). For $\alpha = 0$, the α −plane is the whole FOU($\tilde{A}$).

Now we have to understand the $\boldsymbol{\alpha - cut}$ of the curve $\mu_x(u)$ which is expressed as [8] $S_{\tilde{A}}(x|\alpha) = \{u \in J_x | \mu_x(u) \geq \alpha\} = [S_L(x|\alpha), S_R(x|\alpha)]$, where $S_L(x|\alpha)$ and $S_R(x|\alpha)$ are the end points of the $\alpha - cut$. A visualisation is from [8] here is in figures 5(a) and 5(b).

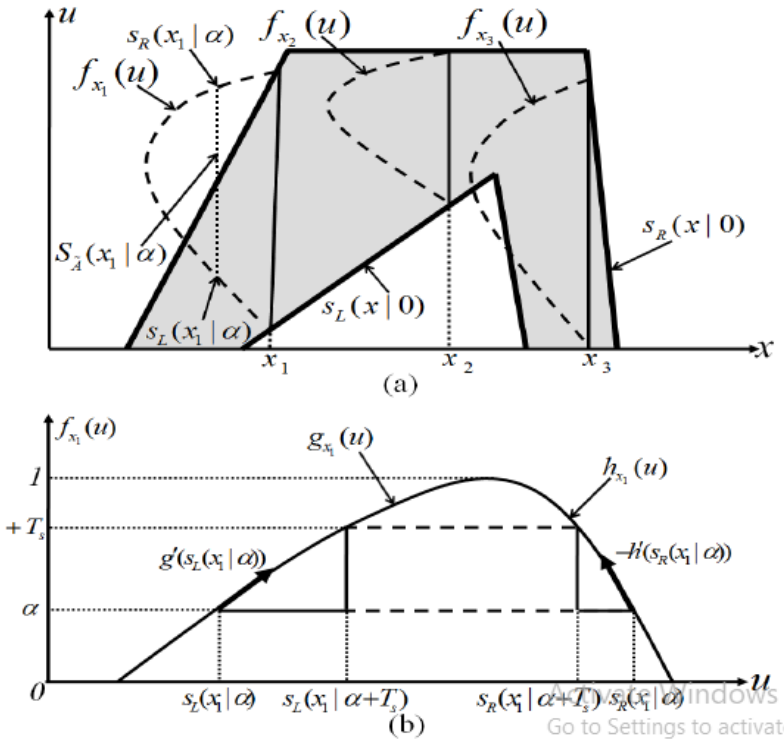


Fig. 5(a): shaded region is FOU of an IT2FS. Fig. 5(b): u-$\mu_x(u)$ plane through the line of J_x

In figure 5(a), the depiction of $S_{\tilde{A}}(x_i|\alpha)$ is slightly wrong as expressed by Zhai and Mendel [8], as definition of $S_{\tilde{A}}(x_i|\alpha)$ says that it is the collection of those u's from J_{x_i} whose membership value is greater than or equal to α, so definitely $S_{\tilde{A}}(x_i|\alpha)$ should lie on J_{x_i} as the projection of $f_{x_i}(u)$ on the $x - u$ plane (since $S_{\tilde{A}}(x_i|\alpha)$ is a line segment on J_{x_i}) and the

curve $\mu_{x_i}(u)$ is here in the picture shown as $f_{x_i}(u)$ which is a 3D curve (lie above J_x along $u - f_{x_i}(u)$ plane) and union of all of such curves (for all i) will make the actual secondary membership function (MF) which is a 3D surface based above the $x - u$ plane. If along every $f_{x_i}(u)$ we imagine a plane that would be the figure 5(b) above. Whatever, for all x_i, the accumulated $S_{\tilde{A}}(x|\alpha)$ is union of $S_{\tilde{A}}(x_i|\alpha)$ and finally making the $\alpha - plane$ $\widetilde{A_\alpha}$. So basically, for an embedded T2FS $\widetilde{A_\alpha}$, $S_R(x|\alpha)$ becomes the UMF and $S_L(x|\alpha)$ becomes the LMF of the FOU($\widetilde{A_\alpha}$). So obviously, when $\alpha = 0$, the corresponding $S_R(x|\alpha)$ and $S_L(x|\alpha)$ becomes the UMF and LMF of the whole FOU($\tilde{A}$).

So the $\alpha - plane\ representation$ of $\tilde{A}$ is

$\tilde{A} = \bigcup_{\alpha\in[0,1]} \alpha/\widetilde{A_\alpha} = \bigcup_{\alpha\in[0,1]} \alpha/\left[\bigcup_{\forall A_e(\alpha)} A_e(\alpha)\right]$. So that we can visualize this as a collection of nested areas (with membership) whose intersection is $\widetilde{A_1}$ and union is $\widetilde{A_0}$.

In CF algorithm, Zhai and Mendel [8] assumed that the secondary MF is convex, which is also a constraint and will be discussed in section 5. hence, $\mu_{x_i}(u)$ is divided into two parts, first one is $g_x(u)$ which is increasing from $S_L(x|0)$ to $S_L(x|1)$ and the second one is $h_x(u)$ decreasing from $S_R(x|1)$ to $S_R(x|0)$. Then using $S_R(x|\alpha)$ as UMF and $S_L(x|\alpha)$ as LMF, centroid of each $\alpha - plane$ is computed by KM algorithm and finally union of all of the centroids is the actual centroid of $\tilde{A}$. But the uniqueness of this algorithm is that KM algorithm is needed for only one $\alpha - plane$ and then using that centroid $c(\tilde{A}_\alpha)$ to its neighbouring $\alpha - plane$ (named as $(\alpha + T_s) - plane$) by some connection equations. These equations actually are relation between $S_R(x|\alpha)$ and $S_R(x|\alpha + T_s)$ and $S_L(x|\alpha)$ and $S_L(x|\alpha + T_s)$ as can be realised from figure 5(b).

$S_L(x|\alpha + T_s) \approx S_L(x|\alpha) + T_s/g'(S_L(x|\alpha))$ and

$S_R(x|\alpha + T_s) \approx S_R(x|\alpha) + T_s/h'(S_R(x|\alpha))$ which are known as $\alpha - plane$ connection equations, where $g'(S_L(x|\alpha))$ is the gradient of $g_x(u)$ at $x = S_L(x|\alpha)$, and

$h'(S_R(x|\alpha))$ is the gradient of $h_x(u)$ at $x = S_R(x|\alpha)$.

Starting the calculation of finding centroid in each $\alpha - plane$ from $\alpha = 0 - plane$ up to $\alpha = 1 - plane$ (where, step-length of α-increament is T_s), they did accumulate collection of $c_l(\widetilde{A_\alpha})$ and $c_r(\widetilde{A_\alpha})$ i.e., collection of centroids $c(\widetilde{A_\alpha}), \alpha = 0, T_s, 2T_s, \dots, 1$. Finally, the centroid $c(\tilde{A}) = \bigcup_{\alpha\in[0,1]} c(\widetilde{A_\alpha})$.

For more details, see section 4 of Zhai and Mendel [8]. But Zhai and Mendel [16] again in 2011, did some tuning on their own work of CF algorithm to eliminate some error generated in each $\alpha -$ plane and named it as Enhanced CF algorithm. In this paper, they started from

median $\alpha = \alpha_M -$ plane to both upwards and downwards so that the accumulating error in CF algorithm could reduce to half.

However, in both of these algorithms, we can find centroids of a prespecified number of $\alpha -$ planes and after aggregating those the final centroid interval comes, which is again considered as a type-1 fuzzy set but no information about the membership function is provided.

4. Proposed Method

Before proposing new method, we want to showcase the different aspect of motivation behind the work.

Motivation 1: Each iterative process developed so far, all of them have expressed the type-reduced set as a T1FS, but we don't know the MF of that set. So, we tried to eliminate the problem of incorporating a new MF to the centroid interval and directly find a crisp point which may be regarded as the centroid. In a summary we are by-passing the two steps of type-reducing & defuzzification with our method. That motivates us to develop such a method, filling the gap.

Motivation 2: The very popular centroid flow algorithm [8] has a limitation which we mentioned earlier that the secondary MF was assumed as a convex set. We want generalization, since in reality any type of surface can truly exist as a secondary MF.

Motivation 3: We looked for a method that could be both computationally easy and adaptable to any form of fuzzy set because the computation complexity of the earlier methods, such as alpha-plane decomposition and many other steps do take a lot of computation time.

Motivation 4: To the best of our knowledge, no publication has been found about the centroid of a type-3 fuzzy set. We will be able to find the centroid in both interval type-3 and general type-3 fuzzy set using our method. Furthermore, we can find centroid in any higher type also (if required). It is noteworthy to highlight that only interval type-3 fuzzy sets (IT3FS) [10], [11], [12], [13], and [14] have been presented as the most recent in research on a variety of issues (not related to centroids); general type-3 fuzzy sets (GT3FS) have not yet been established. Even in 2025, Castillo et al. [12] mentioned that there is no method for calculating centroid yet as he wrote **"…***some theoretical achievement would be needed, like better (faster) type reduction methods and possibly other ways to reduce the number of inferences needed, like through the use of shadow sets. Finally, there is also possible theoretical work to be carried out on general type-n fuzzy logic, with this generalizing the form of the fuzzy sets even more…"*. Mohammadzadeh [15] proposed a type-reduction of

type-3 fuzzy set based on the theorem of uncertainty bounds for IT3FS, but not direct result of centroid.
After developing our strategy, we will eventually show that GT3FS is possible in some instances.

We are proposing a straightforward method to find centroid of fuzzy set. This method easily can be applied to any type, e.g., for type-1 or type-2 or type-3 or higher (if possible, in future). We are going to apply type-1 centroid finding method to every $x-u$ plane corresponding to each x of X in case of type-2 fuzzy set. for type-3 fuzzy set, formula goes as the same way.

For a type-1 fuzzy set A, centroid $c_A = \frac{\sum_{i=1}^{N} x_i u_i}{\sum_{i=1}^{N} u_i}$, where $x_i \in X, u_i \in [0,1]$, as there is no confusion about the method of finding centroid of a T1FS , as pioneered by Zadeh. Although, in every type-reduction method, when a higher type FS is reduced to a centroid interval which is a T1FS, get defuzzified by just averaging two end points of that set, which is against the motive of this paper.

Let us consider a point $x_k (k \in \{1,2,\dots\dots,N\})$ (where N is the number of discretisation of support of the primary variable x) in the type-2 fuzzy set $\tilde{A}$. Hence, if we visualize the $x-u$ plane through the point x_k, then we can see a curve $w = f(u)$, for $u \in J_{x_k}^u \subseteq [0,1]$ (in continuous secondary MF) or $w_j = f(u_j)$ for $u_j \in J_{x_k}^u \subseteq [0,1]$ (in case of discrete secondary MF). If we discretize the line segment connecting (x_k, u_1) and (x_k, u_{M_k}) into M_k points, then centroid of type-1 fuzzy set $\widetilde{U_k}$ is obviously $\frac{\sum_{j=i}^{M_i} u_{kj} w_j}{\sum_{j=1}^{M_1} w_j}$, which can be considered (intuitively) as the weight of the point x_k. So, for a general type-2 fuzzy set $\tilde{A}$, centroid can be defined as

$$c_{\tilde{A}} = \frac{x_1 \frac{\sum_{j=1}^{M_1} u_{1j} w_{1j}}{\sum_{j=1}^{M_1} w_{1j}} + x_2 \frac{\sum_{j=1}^{M_2} u_{2j} w_{2j}}{\sum_{j=1}^{M_2} w_{2j}} + \cdots + x_N \frac{\sum_{j=1}^{M_N} u_{Nj} w_{Nj}}{\sum_{j=1}^{M_N} w_{Nj}}}{\frac{\sum_{j=1}^{M_1} u_{1j} w_{1j}}{\sum_{j=1}^{M_1} w_{1j}} + \frac{\sum_{j=1}^{M_2} u_{2j} w_{2j}}{\sum_{j=1}^{M_2} w_{2j}} + \cdots + \frac{\sum_{j=1}^{M_N} u_{Nj} w_{Nj}}{\sum_{j=1}^{M_N} w_{Nj}}} = \frac{\sum_{i=1}^{N} (x_i \frac{\sum_{j=1}^{M_i} u_{ij} w_{ij}}{\sum_{j=1}^{M_i} w_{ij}})}{\sum_{i=1}^{N} (\frac{\sum_{j=1}^{M_i} u_{ij} w_{ij}}{\sum_{j=1}^{M_i} w_{ij}})} \quad (1)$$

In continuous case, let $f_1(x,u,w)|_{w=0}$ (let $u = f(x)$)and $g_1(x,u,w)|_{w=0}$ (let $u = g(x)$)are upper & lower MF of the general type-2 fuzzy set $\tilde{A}$, and $h_1(x,u,w)$ $(let\ w = h(x,u))$be the secondary MF, then equation (1) will be

$$c_{\tilde{A}} = \frac{\int_{x_1}^{x_N} x \left(\frac{\int_{g(x)}^{f(x)} u.h(x,u)\, du}{\int_{g(x)}^{f(x)} h(x,u)\, du} \right) dx}{\int_{x_1}^{x_N} \left(\frac{\int_{g(x)}^{f(x)} u.h(x,u)\, du}{\int_{g(x)}^{f(x)} h(x,u)\, du} \right) dx} = \frac{\int_{x_1}^{x_N} x \left(\frac{\int_{LMF}^{UMF} u.h(x,u)\, du}{\int_{LMF}^{UMF} h(x,u)\, du} \right) dx}{\int_{x_1}^{x_N} \left(\frac{\int_{LMF}^{UMF} u.h(x,u)\, du}{\int_{LMF}^{UMF} h(x,u)\, du} \right) dx} \quad (2)$$

In case of interval type-2 fuzzy set the formula (1) should be like below.

Since, the secondary membership value is 1 for all $u_j \in J_{x_k}^u$, i.e., $w_{ij} = 1, \forall i \in \{1,2,\dots,N\}, \forall j \in \{1,2,\dots,M_k\}$, therefore, centroid of type-1 fuzzy set $\widetilde{U_k}$ is obviously be $\frac{u_1+u_{M_k}}{2}$. That means, if we focus on whole foot of uncertainty, then to calculating centroid on each $x-u$ plane, it would be

$$\frac{x_1\frac{u_{11}+u_{1M_1}}{2}+x_2\frac{u_{21}+u_{2M_2}}{2}+\cdots+x_N\frac{u_{N1}+u_{N,M_N}}{2}}{\frac{u_{11}+u_{1M_1}}{2}+\frac{u_{21}+u_{2M_2}}{2}+\cdots+\frac{u_{N1}+u_{N,M_N}}{2}} = \frac{\sum_{i=1}^{N} x_i\left(u_{i1}+u_{iM_i}\right)}{\sum_{i=1}^{N}\left(u_{i1}+u_{iM_i}\right)} \tag{3}$$

$$\frac{(x_1u_{11}+x_2u_{21}+\cdots+x_Nu_{N1})+\left(x_1u_{1M_1}+x_2u_{2M_2}+\cdots+x_Nu_{N,M_N}\right)}{(u_{11}+u_{21}+\cdots+u_{N1})+\left(u_{1M_1}+u_{2M_2}+\cdots+u_{N,M_N}\right)}$$

$$=\frac{\int_{x_1}^{x_N} x(LMF+UMF)\ dx}{\int_{x_1}^{x_N}(LMF+UMF)\ dx} \text{ [in continuous case]} \tag{4}$$

This is same as Nie-Tan [27] algorithm which is obviously thought as a special case of our method.

Also from equation (2) we can derive the same equation as we can say that for interval type-2 fuzzy set, $w = h(x,u) = 1$, so $\frac{\int_{g(x)}^{f(x)} u.h(x,u)\ du}{\int_{g(x)}^{f(x)} h(x,u)\ du} = \frac{[u^2/2]_{g(x)}^{f(x)}}{[u]_{g(x)}^{f(x)}} = \frac{f(x)+g(x)}{2}$, so equation (2) becomes $c_{\tilde{A}} = \frac{\int_{x_1}^{x_N} x\left(\frac{f(x)+g(x)}{2}\right)dx}{\int_{x_1}^{x_N}\left(\frac{f(x)+g(x)}{2}\right)dx} = \frac{\int_{x_1}^{x_N} x(f(x)+g(x))\ dx}{\int_{x_1}^{x_N}(f(x)+g(x))\ dx} = \frac{\int_{x_1}^{x_N} x(UMF+LMF)\ dx}{\int_{x_1}^{x_N}(UMF+LMF)\ dx}$, which is same as equation (4).

Now, we can propagate the same technique in case of type-3 fuzzy set also.

Let $\tilde{B}$ is an interval type-3 fuzzy set. now for every $x_k \in \tilde{B}$, there is an interval type-2 fuzzy set $\widetilde{U_k}$. At a point u_{kj} along u-axis, we visualize the $u-w$ plane (where w is the primary membership of U) and calculate the centroid as $\left(\frac{w_{kj,1}+w_{kj,M_{kj}}}{2}\right)$ of the line segment(discretising the line segment at M_{kj} points) connecting $(u_{kj}, w_{kj,1})$ with $(u_{kj},\ w_{kj,M_{kj}})$[since membership value of each $w_{ij,t}(\ for\ t = 1,\dots,M_{ij})$ is unity as $\tilde{B}$ is a interval type-3 fuzzy set] .Then the centroid of the type-2 fuzzy set $\widetilde{U_k}$ is

$$c(\widetilde{U_k}) = \frac{u_{k1}\left(\frac{w_{k1,1}+w_{k1,M_{k1}}}{2}\right)+u_{k2}\left(\frac{w_{k2,1}+w_{k2,M_{k2}}}{2}\right)+\cdots+u_{kM_k}\left(\frac{w_{kM_k,1}+w_{kM_k,M_{kM_k}}}{2}\right)}{\left(\frac{w_{k1,1}+w_{k1,M_{k1}}}{2}\right)+\left(\frac{w_{k2,1}+w_{k2,M_{k2}}}{2}\right)+\cdots+\left(\frac{w_{kM_k,1}+w_{kM_k,M_{kM_k}}}{2}\right)}$$

$$=\frac{\sum_{j=1}^{M_k} u_{kj}\left(\frac{w_{kj,1}+w_{kj,M_{kj}}}{2}\right)}{\sum_{j=1}^{M_k}\left(\frac{w_{kj,1}+w_{kj,M_{kj}}}{2}\right)}$$

Therefore, centroid $c_{\tilde{B}} = \dfrac{\sum_{i=1}^{N} x_i \left(\dfrac{\sum_{j=1}^{M_k} u_{kj} \left(\frac{w_{kj,1}+w_{kj,M_{kj}}}{2} \right)}{\sum_{j=1}^{M_k} \left(\frac{w_{kj,1}+w_{kj,M_{kj}}}{2} \right)} \right)}{\sum_{i=1}^{N} \left(\dfrac{\sum_{j=1}^{M_k} u_{kj} \left(\frac{w_{kj,1}+w_{kj,M_{kj}}}{2} \right)}{\sum_{j=1}^{M_k} \left(\frac{w_{kj,1}+w_{kj,M_{kj}}}{2} \right)} \right)} = \dfrac{\sum_{i=1}^{N} x_i \left(\dfrac{\sum_{j=1}^{M_k} u_{kj} \left(w_{kj,1}+w_{kj,M_{kj}} \right)}{\sum_{j=1}^{M_k} \left(w_{kj,1}+w_{kj,M_{kj}} \right)} \right)}{\sum_{i=1}^{N} \left(\dfrac{\sum_{j=1}^{M_k} u_{kj} \left(w_{kj,1}+w_{kj,M_{kj}} \right)}{\sum_{j=1}^{M_k} \left(w_{kj,1}+w_{kj,M_{kj}} \right)} \right)}$ (5)

So, geometrically, if $f(x_k, u, w)$ and $g(x_k, u, w)$ be two continuous curves indicating the upper and lower MF for the interval type-2 fuzzy set $\widetilde{U_k}$, then

$$c\left(\widetilde{U_k}\right) = \frac{\int_{u_{k1}}^{u_{kM_k}} u \left(\dfrac{f+g}{2} \right) du}{\int_{u_{k1}}^{u_{kM_k}} \left(\dfrac{f+g}{2} \right) du}$$

is the centroid of the line segment connecting (x_k, u_{k1}) and (x_k, u_{kM_k}), (where M_k is the discretisation of the line segment).

But the points $u_{11}, u_{21}, \dots, u_{N1}$ lies on the lower MF (LMF) of the FOU of interval type-2 fuzzy set $\tilde{B}$, whereas the points $u_{1M_1}, u_{2M_2}, \dots, u_{NM_N}$ lies on the upper MF (UMF) of the FOU of $\tilde{B}$.

So, $c_{\tilde{B}} = \dfrac{\int_{x_1}^{x_N} x \left(\dfrac{\int_{LMF}^{UMF} u\left(\frac{f+g}{2}\right) du}{\int_{LMF}^{UMF} \left(\frac{f+g}{2}\right) du} \right) dx}{\int_{x_1}^{x_N} \left(\dfrac{\int_{LMF}^{UMF} u\left(\frac{f+g}{2}\right) du}{\int_{LMF}^{UMF} \left(\frac{f+g}{2}\right) du} \right) dx} = \dfrac{\int_{x_1}^{x_N} x \left(\dfrac{\int_{LMF_1}^{UMF_1} u(UMF2+LMF2) du}{\int_{LMF}^{UMF} (UMF2+LMF2) du} \right) dx}{\int_{x_1}^{x_N} \left(\dfrac{\int_{LMF}^{UMF} u(UMF2+LMF2) du}{\int_{LMF}^{UMF} (UMF2+LMF2) du} \right) dx}$ (6)

Here, we are expressing limits of FOU of for primary membership as [LMF1, UMF1] and limits of FOU for secondary membership as [LMF2, UMF2].

We are now ready to summarise formulas for centroid for different type of fuzzy sets, which is shown in Table-1. In Table-1, column 2 and 4 showing formula for centroid in discrete and continuous case respectively.

If we attempt to generalize the formula for interval type fuzzy sets (either type-2 or type-3), it is simple to understand that in order to find the centroid, we have to first average the last set of LMF and UMF, then putting that as a weight to its previous membership grade to find the weighted average of that grade, and continuing that process, finally find the centroid.

But for generalized type fuzzy sets (either it is GT2FS or GT3FS), the techniques are little different (although GT3FS is yet to develop in literature, we will demonstrate it for calculation purpose). Since for GT3FS, $\frac{w_{kj,1}+w_{kj,M_{kj}}}{2}$ of equation (5) will convert to

$\frac{\sum_{l=1}^{M_{kj}} w_{kj,l} v_{kj,l}}{\sum_{l=1}^{M_{kj}} v_{kj,l}}$, where $v_{kj,l}$ is the weight of the $w_{kj,l}$ i.e., v represents the tertiary membership grade. So, the centroid of a GT3FS be

$$c(\tilde{A}) = \frac{\sum_{i=1}^{N} x_i \left(\frac{\sum_{j=1}^{M_k} u_{kj} \left(\frac{\sum_{l=1}^{M_{kj}} w_{kj,l} v_{kj,l}}{\sum_{l=1}^{M_{kj}} v_{kj,l}} \right)}{\sum_{j=1}^{M_k} \left(\frac{\sum_{l=1}^{M_{kj}} w_{kj,l} v_{kj,l}}{\sum_{l=1}^{M_{kj}} v_{kj,l}} \right)} \right)}{\sum_{i=1}^{N} \left(\frac{\sum_{j=1}^{M_k} u_{kj} \left(\frac{\sum_{l=1}^{M_{kj}} w_{kj,l} v_{kj,l}}{\sum_{l=1}^{M_{kj}} v_{kj,l}} \right)}{\sum_{j=1}^{M_k} \left(\frac{\sum_{l=1}^{M_{kj}} w_{kj,l} v_{kj,l}}{\sum_{l=1}^{M_{kj}} v_{kj,l}} \right)} \right)} \tag{7}$$

In continuous version, this become

$$\frac{\int_{x_1}^{x_N} x \left(\frac{\int_{LMF_1}^{UMF_1} u \left(\frac{\int_{LMF_2}^{UMF_2} wv \, dw}{\int_{LMF_2}^{UMF_2} v \, dw} \right) du}{\int_{LMF}^{UMF} \left(\frac{\int_{LMF_2}^{UMF_2} wv \, dw}{\int_{LMF_2}^{UMF_2} v \, dw} \right) du} \right) dx}{\int_{x_1}^{x_N} \left(\frac{\int_{LMF_1}^{UMF_1} u \left(\frac{\int_{LMF_2}^{UMF_2} wv \, dw}{\int_{LMF_2}^{UMF_2} v \, dw} \right) du}{\int_{LMF}^{UMF} \left(\frac{\int_{LMF_2}^{UMF_2} wv \, dw}{\int_{LMF_2}^{UMF_2} v \, dw} \right) du} \right) dx} \tag{8}$$

We are proposing the name of the method as (weight in all direction) WAD method. Hence, we can put the algorithm to find centroid for interval type and general type differently.

Algorithm for centroid of Interval Type Fuzzy Set (ITFS):

(i) Find average $(cl(i))$ of UMF and LMF of the foot of uncertainty (FOU) of last membership grade.

(ii) Put $cl(i)$ as weight of previous grade to find weighted average $(cl(i-1))$ of that grade.

(iii) Continue process (ii) until the primary membership grade reached, to find $cl(1)$, which is the centroid of the ITFS.

Algorithm for centroid of General Type Fuzzy Set (GTFS):

(i) Find weighted average of last membership grade as $cl(g)$.

(ii) Put $cl(g)$ as the weight of previous grade to find the weighted average $(cl(g-1))$ of that grade.

(iii) Continue process (ii) until the primary membership grade reached, to find $cl(1)$, which is the centroid of the GTFS.

As we can observe that, the first step of the two algorithms is different but same in nature. Because, if we apply the step (i) of GTFS Algorithm to the step (i) of ITFS Algorithm, we can find the same value. So, the GTFS Algorithm is more general in nature.

In future, if General Type Fuzzy Sets of higher type are to develop by chance, we can apply this technique to find centroid similarly. The most advantageous part of this method is that, not only we don't have to develop new technique; we can envisage purpose of higher type fuzzy set according to the algorithm devised here. That is if the last variable of interval fuzzy set of any type changed from fixed value to fuzzified value, the underlying fuzzy set becomes general type. So, in case of GT3FS, it seems that the tertiary variables don't have unit value for all, rather they have fuzzy values. So, instead of homogeneous volume in figure 5(a), we have to consider heterogeneous volume in case of GT3FS. Whatever may be the visualisation of GT3FS, the formula for finding the centroid is always same as equation (8) in case of continuous heterogeneous volume (equation (7) holds for discrete case).

Table 1: Formula of centroid in WAD algorithm for different type fuzzy sets (both in discrete and continuous case)

Type of fuzzy set	**Formula for centroid (Discrete case)**	**Description of variables, functions**	**Formula for centroid (Continuous case)**	**Description of variables, functions**
T1FS (A)	$\frac{\sum_{i=1}^{N} x_i u_i}{\sum_{i=1}^{N} u_i}$	$x_i \in A, u_i$ are the membership value of x_i, $u_i \in [0,1]$	$\frac{\int_{x_1}^{x_N} xu\, dx}{\int_{x_1}^{x_N} u\, dx}$	$x \in A, u$ is membership grade of x
GT2FS ($\tilde{A}$)	$\frac{\sum_{i=1}^{N}(x_i \frac{\sum_{j=1}^{M_i} u_{ij} w_{ij}}{\sum_{j=1}^{M_i} w_{ij}})}{\sum_{i=1}^{N}(\frac{\sum_{j=1}^{M_i} u_{ij} w_{ij}}{\sum_{j=1}^{M_i} w_{ij}})}$	$x_i \in \widetilde{A}$ for $i = 1, \dots, N$, u_{ij}(for $j = 1, \dots, M_i$) is membership value for x_i, w_{ij}(for $i = 1, \dots, N$ and $j = 1, \dots, M_i$) is membership value of u_{ij}	$\frac{\int_{x_1}^{x_N} x \left(\frac{\int_{LMF}^{UMF} u.h(x,u)\ du}{\int_{LMF}^{UMF} h(x,u)\ du} \right) dx}{\int_{x_1}^{x_N} \left(\frac{\int_{LMF}^{UMF} u.h(x,u)\ du}{\int_{LMF}^{UMF} h(x,u)\ du} \right) dx}$	x_1, x_N are two endpoints of $\tilde{A}$, LMF & UMF are lower & upper limit of the FOU for primary membership, $h(x,u)$ is the secondary MF
IT2FS ($\tilde{A}$)	$\frac{\sum_{i=1}^{N} x_i (u_{i1} + u_{iM_i})}{\sum_{i=1}^{N} (u_{i1} + u_{iM_i})}$	$x_i \in \widetilde{A}$ for $i = 1, \dots, N$ u_{i1}, u_{iM_i} are min & max membership value of x_i	$\frac{\int_{x_1}^{x_N} x\,(UMF + LMF)\, dx}{\int_{x_1}^{x_N} (UMF + LMF)\ dx}$	x_1, x_N are two endpoints of $\tilde{A}$, LMF & UMF are lower & upper limit of the FOU for primary membership

GT3FS ($\tilde{A}$)	$\frac{\sum_{i=1}^{N} x_i \left(\frac{\sum_{j=1}^{M_k} u_{kj} \left(\frac{\sum_{l=1}^{M_{kj}} w_{kj,l} v_{kj,l}}{\sum_{l=1}^{M_{kj}} v_{kj,l}} \right)}{\sum_{j=1}^{M_k} \left(\frac{\sum_{l=1}^{M_{kj}} w_{kj,l} v_{kj,l}}{\sum_{l=1}^{M_{kj}} v_{kj,l}} \right)} \right)}{\sum_{i=1}^{N} \left(\frac{\sum_{j=1}^{M_k} u_{kj} \left(\frac{\sum_{l=1}^{M_{kj}} w_{kj,l} v_{kj,l}}{\sum_{l=1}^{M_{kj}} v_{kj,l}} \right)}{\sum_{j=1}^{M_k} \left(\frac{\sum_{l=1}^{M_{kj}} w_{kj,l} v_{kj,l}}{\sum_{l=1}^{M_{kj}} v_{kj,l}} \right)} \right)}$	$x_i \in \widetilde{A}$ for $i = 1, \dots, N$ u_{kj} is the membership value for $x_k, j = 1, \dots, M_k$ $w_{kj,1}$, $w_{kj,M_{kj}}$ are the min & max secondary membership (primary for u_{kj}) $v_{kj,l}$ is tertiary membership grade, $l = 1, \dots, M_{kj}$	$\frac{\int_{x_1}^{x_N} x \left(\frac{\int_{LMF1}^{UMF1} u \left(\frac{\int_{LMF2}^{UMF2} wv \ dw}{\int_{LMF2}^{UMF2} v \ dw} \right) du}{\int_{LMF}^{UMF} \left(\frac{\int_{LMF2}^{UMF2} wv \ dw}{\int_{LMF2}^{UMF2} v \ dw} \right) du} \right) dx}{\int_{x_1}^{x_N} \left(\frac{\int_{LMF1}^{UMF1} u \left(\frac{\int_{LMF2}^{UMF2} wv \ dw}{\int_{LMF2}^{UMF2} v \ dw} \right) du}{\int_{LMF}^{UMF} \left(\frac{\int_{LMF2}^{UMF2} wv \ dw}{\int_{LMF2}^{UMF2} v \ dw} \right) du} \right) dx}$	x_1, x_N are two endpoints of $\tilde{A}$, LMF1 & UMF1 are lower & upper limit of the FOU for primary membership u, LMF2 & UMF2 are lower & upper limit of the FOU for secondary membership w v is the tertiary membership grade
IT3FS ($\tilde{A}$)	$\frac{\sum_{i=1}^{N} x_i \left(\frac{\sum_{j=1}^{M_k} u_{kj} \left(w_{kj,1} + w_{kj,M_{kj}} \right)}{\sum_{j=1}^{M_k} \left(w_{kj,1} + w_{kj,M_{kj}} \right)} \right)}{\sum_{i=1}^{N} \left(\frac{\sum_{j=1}^{M_k} u_{kj} \left(w_{kj,1} + w_{kj,M_{kj}} \right)}{\sum_{j=1}^{M_k} \left(w_{kj,1} + w_{kj,M_{kj}} \right)} \right)}$	$x_i \in \widetilde{A}$ for $i = 1, \dots, N$ u_{kj} is the membership value for $x_k, j = 1, \dots, M_k$ $w_{kj,1}$, $w_{kj,M_{kj}}$ are the min & max secondary membership (primary for u_{kj})	$\frac{\int_{x_1}^{x_N} x \left(\frac{\int_{LMF1}^{UMF1} u(UMF2 + LMF2) \, du}{\int_{LMF}^{UMF} (UMF2 + LMF2) \, du} \right) dx}{\int_{x_1}^{x_N} \left(\frac{\int_{LMF}^{UMF} u(UMF2 + LMF2) \, du}{\int_{LMF}^{UMF} (UMF2 + LMF2) \, du} \right) dx}$	x_1, x_N are two endpoints of $\tilde{A}$, LMF1 & UMF1 are lower & upper limit of the FOU for primary membership u, LMF2 & UMF2 are lower & upper limit of the FOU for secondary membership w

5. Computational Complexity

We will calculate the number of arithmetical operations for each type one by one and summarised in Table 2.

For type-1: N multiplication, $2(N-1)$ additions and 1 division.

For type-2: When numbers of discretisation along x-axis is N and along u-axis is M_i, then for interval type-2, we need $2N$ addition, N multiplication, then again $2(N-1)$ addition, 1 division and for general type-2, we need $2NM_i$ multiplication, $4N(M_i-1)$ addition, 2N divisions, N multiplication, $2N-1$ addition and 1 division.

For type-3: When numbers of discretisation along x-axis is N and along u-axis is M_i and along w-axis is M_{kj}, then for interval type, we need $4NM_k$ addition, $2NM_k$ multiplication, $4N(M_k-1)$ addition, $2N$ division, N multiplication, $2N-1$ addition and finally another 1 division. But for general type-3, we need $4M_{kj}M_kN$ multiplication, $8(M_{kj}-1)M_kN$ addition, $4NM_k$ divisions, $2NM_k$ multiplications, $4N(M_k-1)$ additions, 2N divisions, N multiplication, $2N-1$ addition and 1 division.

So, if $N = M_i = M_{kj}$, then total number of arithmetic operations are

Table 2. Number of computations for calculating centroid of different type fuzzy sets

Type	multiplication	addition	division	Total
T1FS	N	$2(N-1)$	1	$3N-1$
IT2FS	N	$2N+2(N-1) = 2(2N-1)$	1	$5N-1$
GT2FS	$2N^2+N$	4N(N-1) +2N-1 = $4N^2-2N-1$	$2N+1$	$6N^2+N$
IT3FS	$2N^2+N$	$4N^2+4N(N-1)+(2N-1)$ = $8N^2-2N-1$	$2N+1$	$10N^2+N$
GT3FS	$4N^3+2N^2+N$	$8N^2(N-1)+4N(N-1)+2N-1 = 8N^3-4N^2-2N-1$	$4N^2+2N+1$	$12N^3+2N^2+N$

So, we can tell that for general type-d of fuzzy set, then the order of computation will be $O(N^d)$, where N is the amount of discretisation along every axis of d dimensions. For interval type-d fuzzy set, it will be $O(N^{d-1})$.

6. Computational Comparison with Existing Methods

First, we evaluate our method with EKM and EIASC for Interval Type-2 Fuzzy Set. The computing platform is a desktop computer running MATLAB R2024a and Windows 10 64-

bit OS with an Intel core CPU i5-10400F @ 2.90GHz and 8G memory. The computational time was measured by tic & toe function of MATLAB software.

We are taking (for comparison) examples (i), (ii), (iii) below from [4] with discretization of X as $N = 2500$ to $80{,}000$.

(i) **Example 1**: Symmetric Gaussian primary MF, as $X = [0, 10]$, LMF = $\exp\left[-\frac{1}{2}\left(\frac{x-5}{0.25}\right)^2\right]$, UMF = $\exp\left[-\frac{1}{2}\left(\frac{x-5}{1.75}\right)^2\right]$,

(ii) **Example 2**: Symmetric Gaussian primary MF, as X = [0, 10], LMF = $\begin{Bmatrix}\exp\left[-\frac{1}{2}\left(\frac{x-7.5}{1}\right)^2\right], x \le 4.98 \\ \exp\left[-\frac{1}{2}\left(\frac{x-2.4}{1}\right)^2\right], x > 4.98\end{Bmatrix}$, UMF = $\begin{Bmatrix}exp\left[-\frac{1}{2}\left(\frac{x-2.4}{1}\right)^2\right], x < 2.4 \\ 1,\ 2.4 \le x \le 7.5 \\ \exp\left[-\frac{1}{2}\left(\frac{x-7.5}{1}\right)^2\right], x > 7.5\end{Bmatrix}$

(iii) **Example 3**: Non-symmetrical FOU whose LMF is non-symmetrical triangular and UMF is non-symmetrical Gaussian, i.e.,

LMF = $\begin{Bmatrix}\frac{0.6(X+5)}{19}, & x \in [-5, 2.6] \\ \frac{0.4(14-x)}{19}, & x \in [2.6, 14]\end{Bmatrix}$,

UMF = $\begin{Bmatrix}exp\left[-\frac{1}{2}\left(\frac{x-2}{5}\right)^2\right], x \in [-5, 7.185] \\ \exp\left[-\frac{1}{2}\left(\frac{x-9}{1.75}\right)^2\right], x \in [7.185, 14]\end{Bmatrix}$

Results for example 1:

Table 3a: Value of centroids according to EKM Algorithm, EIASC Algorithm and the proposed method WAD Algorithm

N	$100*$	2500	5000	10000	20000	40000	80000
EKM	3.8612	3.8795	3.8799	3.8801	3.8802	3.8802	3.8802
EIASC	3.8612	3.8795	3.8799	3.8801	3.8802	3.8802	3.8802
WAD	3.8737	3.8918	3.8922	3.8923	3.8924	3.8925	3.8925

Although we are type-reducing the T2FS to a crisp output in a single step, the value of the centroid is slightly different because, as we previously mentioned, the problem persisted in any of the iterative algorithms developed so far in the final step when taking the average of two end-points of the centroid interval, since that is not the type-reduction technique literally. Here, in the horizontal axis, units are according to column N (number of discretization) of Tables 3a and 3b.

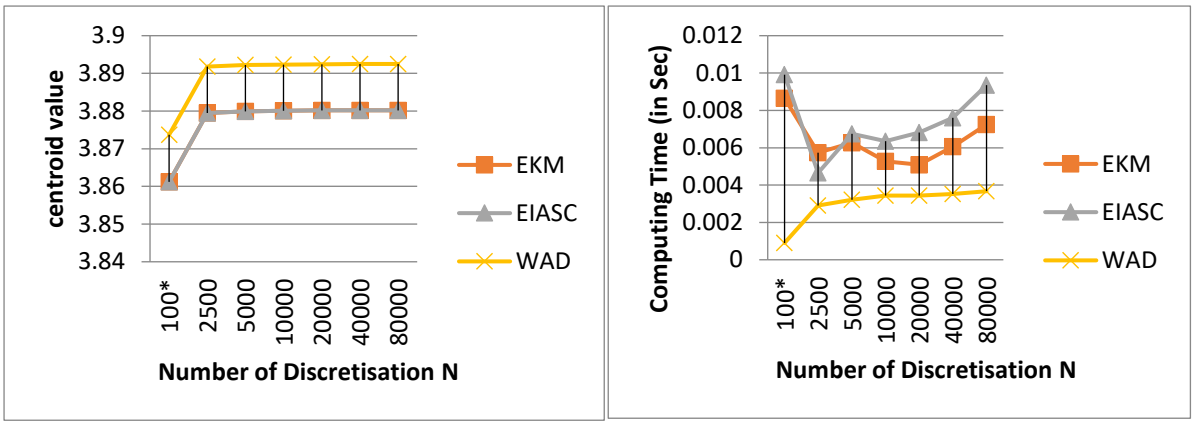


Fig. 6: Visualisation of Table 3a and 3b

Table 3b: Computation time (in sec) elapsed corresponding to the Table 3a (*no of discretisation = 100 is taken for future purpose)

N	$100*$	2500	5000	10000	20000	40000	80000
EKM	0.00865	0.00574	0.00626	0.00528	0.00509	0.00606	0.00725
EIASC	0.00991	0.00465	0.00675	0.00635	0.00682	0.00761	0.00935
WAD	0.0009	0.00291	0.00321	0.00344	0.00343	0.00353	0.00367

Observe that with discretization, increase in computation time in each step in previous algorithms is more than 4 times (on an average) than WAD-algorithm as illustrated in figure 6. Here, we can also see that, on average, our approach saved nearly 47% of the time compared to earlier methods.

Results for example 2:

Table 4a: Value of centroids according to EKM Algorithm, EIASC Algorithm and the proposed method WAD Algorithm

N	2500	5000	10000	20000	40000
EKM	3.614	3.6143	3.6145	3.6146	3.6146
EIASC	3.614	3.6143	3.6145	3.6146	3.6146
WAD	3.8783	3.8786	3.8787	3.8788	3.8788

Here, in the horizontal axis, units are according to column N (number of discretization) of Tables 4a and 4b.

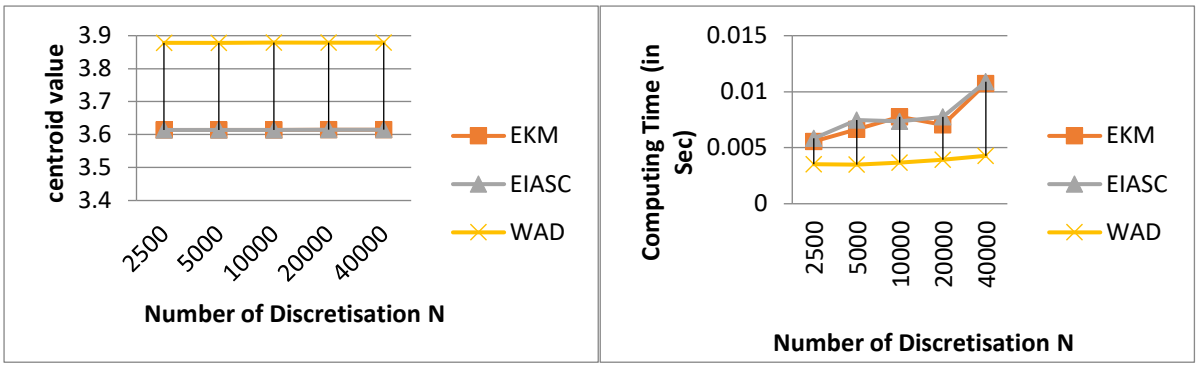


Fig. 7: Visualisation of Tables 4a and 4b

Similar observations to those as example 1 have also been made here.

Table 4b: Computation time (in sec) elapsed corresponding to the Table 4a

N	2500	5000	10000	20000	40000
EKM	0.00556	0.00664	0.00776	0.00703	0.01072
EIASC	0.00583	0.00748	0.00735	0.00777	0.01089
WAD	0.00352	0.00349	0.00366	0.00393	0.00428

Observe that with discretization, increase in computation time in each step in previous algorithms is more than 6.7 times (on an average) than WAD-algorithm as showed in figure 7. A comparison in computing time of our algorithm with previous 3 algorithms is displayed graphically as well. Here, we can see that our approach saved an average of about 53% of the time compared to earlier methods.

Results for example 3:

Table 5a: Value of centroids according to EKM Algorithm, EIASC Algorithm and the proposed method WAD Algorithm

N	2500	5000	10000	20000	40000
EKM	5.3613	5.3611	5.361	5.3609	5.3609
EIASC	5.3613	5.3611	5.361	5.3609	5.3609
WAD	5.5426	5.5424	5.5422	5.5421	5.5421

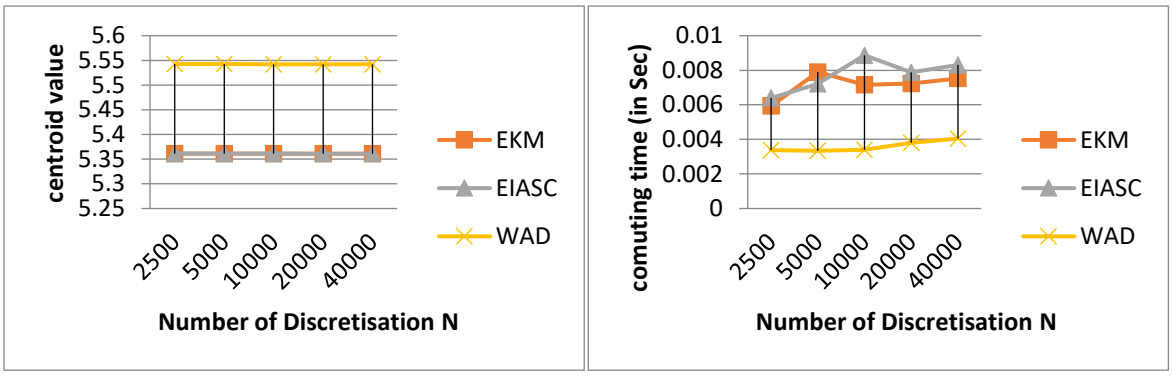


Fig. 8: Visualisation of Tables 5a and 5b

Table 5b: Computation time (in sec) elapsed corresponding to the Table 5a is shown here

N	2500	5000	10000	20000	40000
EKM	0.00593	0.0079	0.00715	0.00723	0.00753
EIASC	0.00639	0.00722	0.00887	0.00788	0.00831
WAD	0.00337	0.00334	0.00339	0.0038	0.00405

Figure 8 illustrates how discretization increases the processing time in each step of earlier algorithms by an average of nearly 6.3 times that of the WAD algorithm. A comparison in computation time of our algorithm with previous 3 algorithms is shown graphically. Here, in the horizontal axis, units are according to column N (number of discretization) of Table 5a and 5b. Here, also we can see that our approach saved an average of about 53% of the time compared to earlier methods.

So, these were demonstration for interval type-2 fuzzy sets. Now we will give some examples for General Type-2 Fuzzy sets.

Example 4: We will take primary membership as Symmetric Gaussian primary MF, UMF = $\exp\left[-\frac{1}{2}\left(\frac{x-4}{1.75}\right)^2\right]$, LMF = $0.25 * \exp\left[-\frac{1}{2}\left(\frac{x-6}{0.25}\right)^2\right]$, then at any point x of X, the secondary MF is $\exp\left[-\frac{1}{2}\left(\frac{u-m_u}{\sigma_u}\right)^2\right]$, where $m_u = (\text{UMF} + \text{LMF})/2$, $\sigma_u = (\text{UMF} - \text{LMF})/4$ and $X = [0, 10]$. This GT2FS is depicted in figure 9. We have showed the value of the crisp centroid of this set in Tables 6a and 6b.

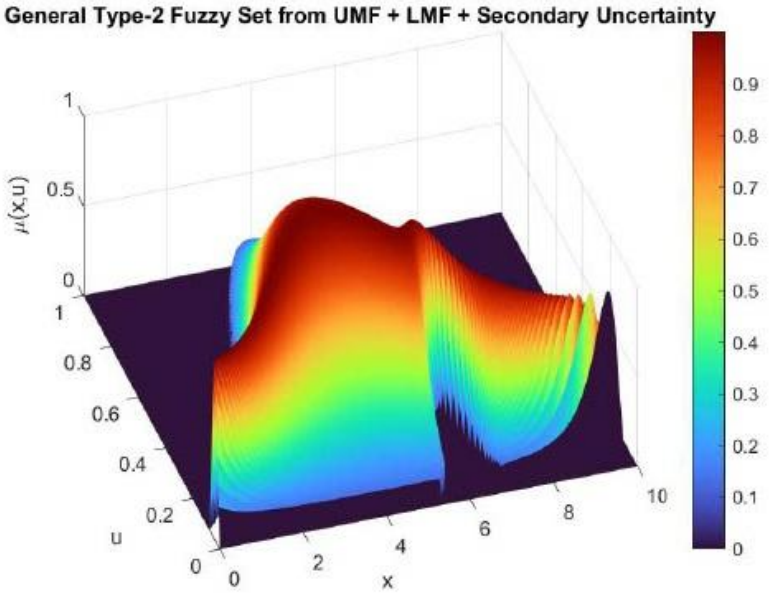


Fig. 9: GT2FS of example 4

Now according to our proposed method, the formula for centroid of a general type-2 fuzzy set is

$$\frac{\int_{x_1}^{x_N} x\left(\frac{\int_{LMF}^{UMF} u.\,h(x,u)\ du}{\int_{LMF}^{UMF} h(x,u)\ du}\right) dx}{\int_{x_1}^{x_N}\left(\frac{\int_{LMF}^{UMF} u.\,h(x,u)\ du}{\int_{LMF}^{UMF} h(x,u)\ du}\right) dx}$$

Here, in our example, $x_1 = 0,\ x_N = 10,\ UMF = \exp\left[-\frac{1}{2}\left(\frac{x-4}{1.75}\right)^2\right],\ LMF = 0.25 * \exp\left[-\frac{1}{2}\left(\frac{x-6}{0.25}\right)^2\right],\ h(x,u) = \exp\left[-\frac{1}{2}\left(\frac{u-(UMF+LMF)/2}{(UMF-LMF)/4}\right)^2\right]$

Therefore $h(x,u) = \exp\left[-\frac{1}{2}\left(\frac{u-\left\{\left(\exp\left[-\frac{1}{2}\left(\frac{x-5}{1.75}\right)^2\right]\right)+\left(0.25*\exp\left[-\frac{1}{2}\left(\frac{x-5}{0.25}\right)^2\right]\right)\right\}/2}{\left\{\left(\exp\left[-\frac{1}{2}\left(\frac{x-5}{1.75}\right)^2\right]\right)-\left(0.25*\exp\left[-\frac{1}{2}\left(\frac{x-5}{0.25}\right)^2\right]\right)\right\}/4}\right)^2\right]$

Table 6a: Value of centroids of example 4 according to our proposed method WAD Algorithm

N	10	20	40	80	100	120
WAD	4.16154	4.10587	4.08527	4.08452	4.08449	4.08448

Table 6b: Computation time elapsed corresponding to the WAD Algorithm

N	10	20	40	80	100	120
Time (s)	0.004280	0.007381	0.001369	0.025993	0.032493	0.037926

In contrast to the interval type-2 scenario, we find that the convergence rate is extremely high in relation to the number of discretizations. We provide an example from [20] to compare our approach with the IEODS algorithm. Details are as follows:

The LMF $= max\{0.5exp\left(-\frac{(x-3)^2}{4}\right), 0.4\exp\left(-0.5\frac{(x-6)^2}{4}\right)\}$

UMF $= max\{exp\left(-\frac{(x-3)^2}{2}\right), 0.8\exp\left(-0.5\frac{(x-6)^2}{2}\right)\}$, $X \in [0,10]$.

The secondary membership function is a trapezoid with unit height, whose base equal to $UMF - LMF$, and top is defined by two end points as($0.4 * LMF + 0.6 * UMF$) and ($0.4 * UMF + 0.6 * LMF$) respectively. Results are shown in Tables 7a and 7b.

Table 7a: Value of centroids of example 4 according to our proposed method WAD Algorithm

N	5	10	20	50	100	1000
WAD	4.380416	4.332437	4.329266	4.328109	4.328029	4.32798

Table 7b: Computation time elapsed corresponding to the Table 7a

N	5	10	20	50	100	1000
WAD	**0.02993**	**0.05495**	**0.10415**	**0.25328**	**0.490744**	**4.8265**
IEODS	0.64(α=20)	0.65(α=20)	0.66(α=20)	0.71(α=20)	0.77(α=20)	8.89(α=100)

The second column of Table 7a is kept blank because Wu et al. [20] did not display the value of the centroid interval left end or right end point. Observe in Table 7b that, computation time is very low in our method both in the case of 20 α-planes and 100 α-planes, although they computed on a higher capacity machine which was Intel Core CPU i7-4790 @ 3.60 Hz and 24G memory, running on windows 10 64-bit Enterprise whereas we are using Intel core CPU i5-10400F @ 2.90GHz and 8G memory, running windows 10 64-bit OS.

Centroid for Type-3 Fuzzy Set:

Centroid calculation for type-3 fuzzy sets is a long-awaited topic. While type-3 control has been the focus of some research articles, type-3 fuzzy system type-reduction utilizing centroid has not received the same level of attention as type-2 FLS. We take example 4 into consideration when calculating the centroid for a type-3 fuzzy set, but there also be some differences. In earlier scenario, the secondary membership grade was represented by a single 3D surface.

However, in the case of IT3FS, as shown in figure 10(a), the secondary membership grade will be represented by a volume, which can be compared to a bundle of surfaces. Additionally, the FOU of an IT3FS differs slightly from that of an IT2FS or GT2FS. Here, in figure 10(b), the black shaded zone is FOU of the lower secondary membership grade and the red shaded region is FOU of the higher secondary membership grade. Therefore, we have to consider an extra surface in addition to example 4 above. Let us write the whole consideration again from scratch.

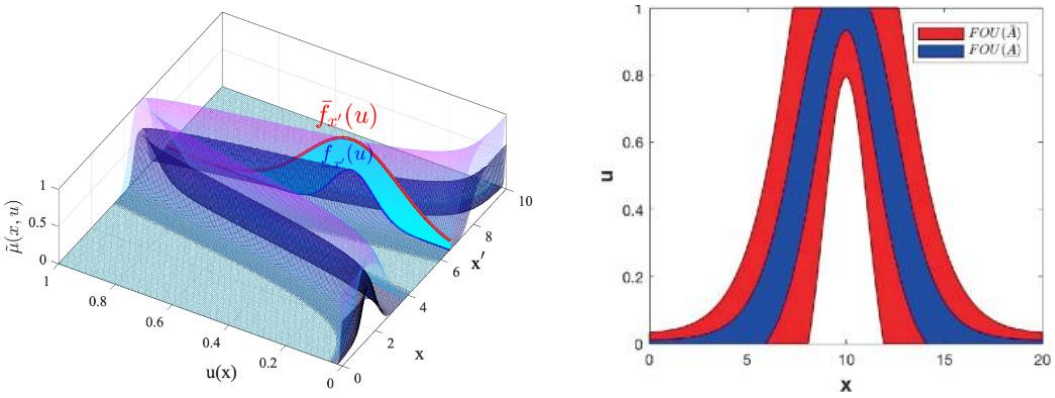


Fig. 10(a): graphical representation of an IT3FS (taken from [17]) Fig. 10(b): FOU of an IT3FS (taken from [18])

Example 5: We will take primary membership as Symmetric Gaussian primary MF, UMF1 = $\exp\left[-\frac{1}{2}\left(\frac{x-4}{1.75}\right)^2\right]$, LMF1 = $0.25 * \exp\left[-\frac{1}{2}\left(\frac{x-6}{0.25}\right)^2\right]$, then at any point x of X, the secondary UMF is UMF2 = $\exp\left[-\frac{1}{2}\left(\frac{u-m_u}{\sigma_u}\right)^2\right]$, secondary LMF is LMF2 = $0.5 * \exp\left[-\frac{1}{2}\left(\frac{u-m_u}{0.5*\sigma_u}\right)^2\right]$ where $m_u = (\text{UMF1} + \text{LMF1})/2$, $\sigma_u = (\text{UMF1} - \text{LMF1})/4$ and $X = [0, 10]$. For the visualisation of this example see figure 11. Here, in this example,

$$\mathrm{x}_1 = 0,\ \mathrm{x}_N = 10, \text{UMF1} = \exp\left[-\frac{1}{2}\left(\frac{x-4}{1.75}\right)^2\right], \text{LMF1} = 0.25 * \exp\left[-\frac{1}{2}\left(\frac{x-6}{0.25}\right)^2\right],$$

$$\text{UMF2} = \exp\left[-\frac{1}{2}\left(\frac{u-(\text{UMF1}+\text{LMF1})/2}{(\text{UMF1}-\text{LMF1})/4}\right)^2\right] = \exp\left[-\frac{1}{2}\left(\frac{u-(\exp\left[-\frac{1}{2}\left(\frac{x-4}{1.75}\right)^2\right]+0.25*\exp\left[-\frac{1}{2}\left(\frac{x-6}{0.25}\right)^2\right])/2}{(\exp\left[-\frac{1}{2}\left(\frac{x-4}{1.75}\right)^2\right]-0.25*\exp\left[-\frac{1}{2}\left(\frac{x-6}{0.25}\right)^2\right])/4}\right)^2\right]$$

$$\text{LMF2} = 0.5 * \exp\left[-\frac{1}{2}\left(\frac{u-(\exp\left[-\frac{1}{2}\left(\frac{x-4}{1.75}\right)^2\right]+0.25*\exp\left[-\frac{1}{2}\left(\frac{x-6}{0.25}\right)^2\right])/2}{0.5*(\exp\left[-\frac{1}{2}\left(\frac{x-4}{1.75}\right)^2\right]-0.25*\exp\left[-\frac{1}{2}\left(\frac{x-6}{0.25}\right)^2\right])/4}\right)^2\right].$$

Results of this example are shown in Tables 8a and 8b.

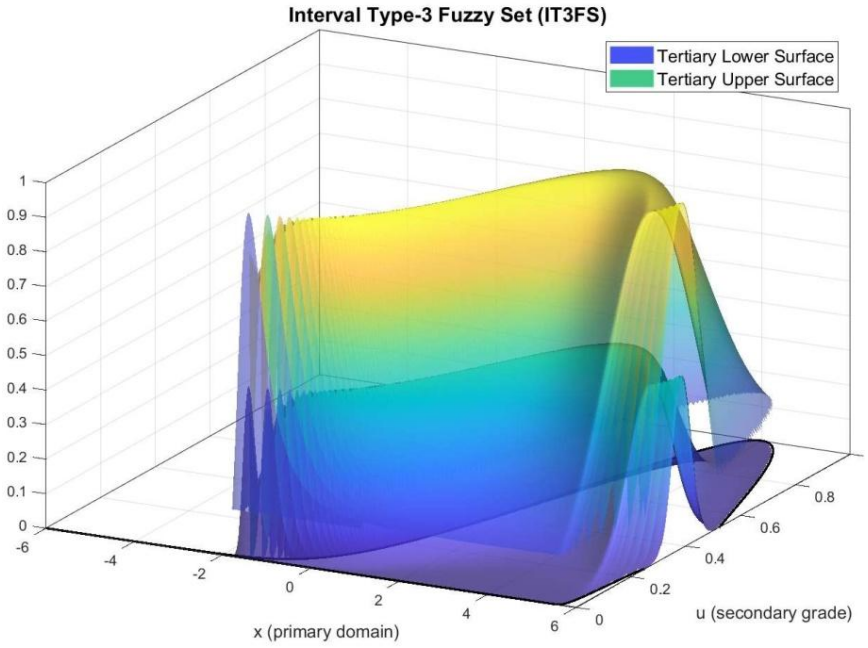


Fig 11: IT3FS of example 5

The formula for computing centroid of an IT3FS is

$$\frac{\int_{x_1}^{x_N} x\left(\frac{\int_{LMF1}^{UMF1} u(UMF2+LMF2)\,du}{\int_{LMF}^{UMF}(UMF2+LMF2)\,du}\right)dx}{\int_{x_1}^{x_N}\left(\frac{\int_{LMF}^{UMF} u(UMF2+LMF2)\,du}{\int_{LMF}^{UMF}(UMF2+LMF2)\,du}\right)dx}$$

The IT3FS centroid computation time is shown below:

Table 8a: Value of centroids of example 5 according to our proposed WAD Algorithm

N	10	20	40	80	100	120
WAD	4.16154	4.10587	4.08527	4.08452	4.08449	4.08448

Table 8b: Computation time elapsed corresponding to the WAD algorithm

N	10	20	40	80	100	120
Time (s)	0.004364	0.007858	0.014700	0.028171	0.03469	0.040527

Value of the centroid is getting converged in 4 decimal places after almost near 100-point discretisation. Comparison with other existing methods is not possible as those are applicable only up to IT2FS. We have selected the example as a random one, because our formula does not depend on the nature of the curve or surface corresponding to any membership grade.

Any non-symmetric, non-convex curve or surface can be used by the researchers as long as it is appropriate.

7. Conclusion

We developed a centroid method for higher order fuzzy sets after showcasing other existing methods. It is observed that instead of different computational complexity for different type fuzzy set we can maintain continuity in the proposed method. We can use the proposed approach as already demonstrated the formula for general type-3 fuzzy sets. Theoretically, it might work with type-n ($n > 3$) fuzzy sets. We have demonstrated how general type-3 fuzzy set can be thought of as a geometric representation in section 4.

Compared to earlier approaches, our method has demonstrated superior computing speed in both interval type-2 and general type-2 fuzzy sets. We have also shown the direct formula for centroid as a precise value for type-3 fuzzy sets. When finding the centroid is the main goal in a fuzzy logic system (FLS), the need for defuzzification following type-reduction remain superfluous because we can now determine the crisp centroid in a single calculation from any higher order fuzzy sets.

Since we employed a straightforward trapezoidal integration technique for the various calculations in Table 1, future scope of this study may include rapid integration strategies. It will be very beneficial for engineers to use this novel WAD method in control theory since centroid plays a significant and elaborative role in the control parameter design process.

References

1. Karnik, N.N., Mendel, J.M., 2001. Centroid of a type-2 fuzzy set. *Information Sciences* 132 (1–4), 195–220. https://doi.org/10.1016/S0020-0255(01)00069-X
2. Wu, D., Mendel, J.M., 2009. Enhanced Karnik–Mendel algorithms. *IEEE Transactions on Fuzzy Systems* 17 (4), 923–934. https://doi.org/10.1109/TFUZZ.2008.924329
3. Durán, K., Bernal, H., Melgarejo, M., 2008. Improved iterative algorithm for computing the generalized centroid of an interval type-2 fuzzy set. In: *Proc. NAFIPS 2008*, New York, USA, pp. 1–5. https://doi.org/10.1109/NAFIPS.2008.4531244
4. Mendel, J.M., Liu, F., 2007. Super-exponential convergence of the Karnik–Mendel algorithms for computing the centroid of an interval type-2 fuzzy set. *IEEE Transactions on Fuzzy Systems* 15 (2), 309–320. https://doi.org/10.1109/TFUZZ.2006.882463
5. Mendel, J.M., Hagras, H., Tan, W.W., Melek, W.W., Ying, H., 2014. Introduction. In: *Introduction to Type-2 Fuzzy Logic Control: Theory and Applications*. IEEE Press, pp. 1–31. https://doi.org/10.1002/9781118886540

6. Mendel, J.M., John, R.I.B., Type-2 fuzzy sets made simple. *IEEE Transactions on Fuzzy Systems* 10 (2), 117–127, 2002. https://doi.org/10.1109/91.995115

7. Mendel, J.M., Liu, F., Zhai, D., α-Plane representation for type-2 fuzzy sets: Theory and applications. *IEEE Transactions on Fuzzy Systems* 17 (5), 1189–1207. 2009. https://doi.org/10.1109/TFUZZ.2009.2024411

8. Zhai, D., Mendel, J.M., 2010. Centroid of a general type-2 fuzzy set computed by means of the centroid-flow algorithm. In: *Proc. IEEE Int. Conf. on Fuzzy Systems*, Barcelona, Spain, pp. 1–8. https://doi.org/10.1109/FUZZY.2010.5584547

9. Mizumoto, M., Tanaka, K., 1981. Fuzzy sets and type-2 under algebraic product and algebraic sum. *Fuzzy Sets and Systems* 5 (3), 277–290. https://doi.org/10.1016/0165-0114(81)90056-7

10. Castillo, O., Valdez, F., Melin, P., Ding, W., 2024. A survey on type-3 fuzzy logic systems and their control applications. *IEEE/CAA Journal of Automatica Sinica* 11 (8), 1744–1756. https://doi.org/10.1109/JAS.2024.124530

11. Valdez, F., Castillo, O., Melin, P., 2025. A bibliometric review of type-3 fuzzy logic applications. *Mathematics* 13 (3), 375. https://doi.org/10.3390/math13030375

12. Mohammadzadeh, A., Castillo, O., Band, S.S., Mosavi, A., 2021. A novel fractional-order multiple-model type-3 fuzzy control for nonlinear systems with unmodeled dynamics. *International Journal of Fuzzy Systems* 23, 1633–1651. https://doi.org/10.1007/s40815-021-01063-3

13. Mohammadzadeh, A., Sabzalian, M.H., Zhang, W., 2020. An interval type-3 fuzzy system and a new online fractional-order learning algorithm: Theory and practice. *IEEE Transactions on Fuzzy Systems* 28, 1940–1950. https://doi.org/10.1109/TFUZZ.2019.2946556

14. Mohammadzadeh, A., Vu, M.T., Taghavifar, H., et al., 2025. A type-3 fuzzy logic system with uncertainty-bound type-reduction and optimized secondary memberships and α-cut levels. *International Journal of Fuzzy Systems*. https://doi.org/10.1007/s40815-025-02082-1

15. Zhai, D., Mendel, J.M., 2011. Enhanced centroid-flow algorithm for general type-2 fuzzy sets. In: *Proc. NAFIPS 2011*, El Paso, TX, USA, pp. 1–6. https://doi.org/10.1109/NAFIPS.2011.5751918

16. Peraza, C., Ochoa, P., Castillo, O., Geem, Z.W., 2022. Interval type-3 fuzzy differential evolution for designing an interval type-3 fuzzy controller of a unicycle mobile robot. *Mathematics* 10, 3533. https://doi.org/10.3390/math10193533

17. Castillo, O., Castro, J.R., Melin, P., 2022. Interval type-3 fuzzy sets. In: *Interval Type-3 Fuzzy Systems: Theory and Design*. Studies in Fuzziness and Soft Computing, vol. 418. Springer, Cham. https://doi.org/10.1007/978-3-030-96515-0_3

18. Mendel, J.M., 2017. *Uncertain Rule-Based Fuzzy Systems*. Springer, Cham. https://doi.org/10.1007/978-3-319-51370-6

19. Wu, L., Qian, F., Wang, L., Ma, X., 2022. An improved type-reduction algorithm for general type-2 fuzzy sets. *Information Sciences* 593, 99–120. https://doi.org/10.1016/j.ins.2022.01.078

20. Wu, H., Mendel, J.M., 2002. Uncertainty bounds and their use in the design of interval type-2 fuzzy logic systems. *IEEE Transactions on Fuzzy Systems* 10 (5), 622–639. https://doi.org/10.1109/TFUZZ.2002.803496

21. Mendel, J.M., Wu, H., 2006. Type-2 fuzzistics for symmetric interval type-2 fuzzy sets: Part 1. Forward problems. *IEEE Transactions on Fuzzy Systems* 14 (6), 781–792.

22. Hu, H.Z., Zhao, G., Yang, H.N., 2010. Fast algorithm to calculate generalized centroid of interval type-2 fuzzy set. *Control and Decision* 25 (4), 637–640.

23. Hu, H., Wang, Y., Cai, Y., 2012. Advantages of the enhanced opposite-direction searching algorithm for computing the centroid of an interval type-2 fuzzy set. *Asian Journal of Control* 14 (5), 1422–1430. https://doi.org/10.1002/asjc.477

24. Celemin, C.E., Melgarejo, M.A., 2012. A faster iterative computation of the centroid of an interval type-2 fuzzy set. In: *Proc. NAFIPS 2012*, Berkeley, CA, USA, pp. 1–6. https://doi.org/10.1109/NAFIPS.2012.6291007

25. Melgarejo, M., 2007. A fast recursive method to compute the generalized centroid of an interval type-2 fuzzy set. In: *Proc. NAFIPS 2007*, San Diego, CA, USA, pp. 190–194. https://doi.org/10.1109/NAFIPS.2007.383835

26. Wu, D., Nie, M., 2011. Comparison and practical implementation of type-reduction algorithms for type-2 fuzzy sets and systems. In: *Proc. IEEE Int. Conf. on Fuzzy Systems*,Taipei, Taiwan, pp. 2131–2138.https://doi.org/10.1109/FUZZY.2011.6007317

27. Nie, M., Tan, W.W., 2008. Towards an efficient type-reduction method for interval type-2 fuzzy logic systems. In: *Proc. IEEE Int. Conf. on Fuzzy Systems*, Hong Kong, China, pp. 1425–1432. https://doi.org/10.1109/FUZZY.2008.4630559

28. Begian, M.B., Melek, W.W., Mendel, J.M., 2011. On the robustness of type-1 and interval type-2 fuzzy logic systems in modelling. *Information Sciences* 181 (7), 1325–1347. https://doi.org/10.1016/j.ins.2010.11.003

29. Begian, M.B., Melek, W.W., Mendel, J.M., 2010. On the stability of interval type-2 TSK fuzzy logic control systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 40 (3), 798–818. https://doi.org/10.1109/TSMCB.2009.2029986

30. Begian, M.B., Melek, W.W., Mendel, J.M., 2008. Stability analysis of type-2 fuzzy systems. In: *Proc. IEEE Int. Conf. on Fuzzy Systems*, Hong Kong, China, pp. 947–953. https://doi.org/10.1109/FUZZY.2008.4630483

31. Zadeh, L.A., 1965. Fuzzy sets. *Information and Control* 8 (3), 338–353. https://doi.org/10.1016/S0019-9958(65)90241-X