Acta Cybernetica **00** (0000) 1–20.

Computing equivalent affinity classes in a fuzzy connectedness framework

Gergely Gulyás, [*] József Dombi [†]

Abstract

The equivalence of affinities in fuzzy connectedness (FC) is a novel concept which gives us the ability to study affinity functions and their precise connection with FC algorithms. Two seminal papers [10, 11] create a strong theoretical background and provide some useful practical examples. Our intention here is to investigate this concept further, because from a practical viewpoint, if we are able to determine the equivalence classes for a given set of affinity functions and narrow it down to a much smaller set of nonequivalent affinities, then the set can be used more effectively in an optimization framework which searches for the best affinity function or parameters for a special task. In other words, we can find the best configuration for a set of given hardware or an image set with special characteristics. From a theoretical perspective, we are interested in the complexity of this problem, i.e. determining equivalence classes. Here, an affinity operator is used which is a function of a given parameter and maps different parameter values for different affinity functions. Our first questions, namely how many different meaningful, nonequivalent affinities there are and how we can enumerate them, led us to a general problem of how the equivalent affinities partition the parameter's domain and how the corresponding equivalence classes can be determined. We will provide a general algorithm schema to construct special algorithms which are able to compute the equivalence classes. We will also analyze a special but very common scenario of when the affinity operator combines two affinities (e.g. a homogeneity and an object feature based affinity) using an aggregation operator (e.g. weighted average) and the particular parameter defines the weights of the affinities. Based on the general algorithm schema, we propose algorithms for this special case and we determine their complexity as well. These algorithms are tested on two sets of medical images, namely, 25 digital dermoscopy images 1280×1024 pixels in size and 3×25 simulated brain MRI slices 181×217 in size.

Keywords: Fuzzy Connectedness, Affinity Functions, Equivalence of Affinities, Image Segmentation, Equivalence of algorithms

[*]E-mail: `gulyasg@inf.u-szeged.hu, gergely.gulyas.uni@gmail.com`
[†]E-mail: `dombi@inf.u-szeged.hu`

1 Introduction

In general, the goal of image segmentation is to partition the image into meaningful object regions. However, this is one of the most difficult tasks in the image processing realm with numerous open questions. Many different approaches exist to handle this problem, one of the most popular being the family of region-based algorithms [3, 4, 32, 34] where the objects are described by their filled regions.

Fuzzy techniques are also widely used in image processing [5, 6, 7, 19, 26, 33], due to the fact that they address the problem of ambiguity in digital images (caused by noise or imprecision, for example). Fuzzy connectedness (FC) [12, 23, 29, 30, 34, 35, 38] is a region-based, fuzzy segmentation framework that has good theoretical support and has been used successfully in several medical applications [21, 22, 28, 31, 36].

In the FC framework, a global fuzzy relation, called fuzzy connectedness, characterizes how the image elements hang together to build up objects. The strength of this relation between any two image elements (or spels) c and d, which refers to the strength of their connectedness, can be determined in the following way. Consider all the possible paths connecting c and d. Each path is a sequence of spels, starting from c and ending in d, with the successive spels being nearby. Each consecutive spel pair constitutes a link and we assign a strength to every path, which is the strength of the weakest link along the path. The strength of connectedness between c and d is the strength of the strongest path between c and d. A local fuzzy relation, called affinity function, is used to determine the strength of the consecutive spel pairs (i.e. the strength of links). Generally speaking, the strength of affinity between any two spels depends on how close the spels are spatially and how similar their properties (like intensity and colour) are in the image.

FC algorithms are parametric in nature. This means that object feature based affinities require some a priori knowledge about the object (e.g. expected mean and standard deviation of the intensities [23]), while object feature-based and homogeneity affinities are combined in many scenarios where the aggregation operator requires some weights [35]. One of the most challenging problems with parametric algorithms in real applications is to find the optimal parameter values, because a given solution can only be evaluated by human observation and cannot be automated. In addition, the parameter domain (i.e. the search space, which may be large or infinite) and the running time of the particular algorithm can also limit the speed of testing. Equivalence of affinities [10, 11], which is a novel concept in the FC framework, gives us the ability to address this problem. Informally, if two affinities used in the same FC schema are equivalent, they lead to identical segmentations [10]. Accordingly, if we could filter the redundant, equivalent affinities from our experimental set, it would definitely reduce the search space. Here, we focus on the theoretical background and algorithmic questions of the former, namely how many different meaningful, non-equivalent affinities there are and how we can enumerate them; or more generally, how the equivalence classes can be characterized. In our model scenario, we have an affinity operator (Sec. 2.2) that is a function of a given parameter, so it maps different parameter values for different affinities. For example, suppose we have two affinities κ_1 and κ_2 and we combine them into κ_w using the weighted average operator with the parameter w:

$$\kappa_w = w\kappa_1 + (1 - w)\kappa_2.$$

In this study, we propose using a general algorithm schema to create special algorithms which are able to determine the set of equivalence classes based on the parameter value of the affinity operator. Then, we investigate a very common scenario where two affinities are combined by means of a weighted quasi-linear mean [18] (e.g. weighted average as in the example above) and some concrete algorithms are built based on the general algorithm schema. The complexity of these algorithms are also considered, and we will show that the structure of equivalence classes is very simple in this case. Lastly, we test the algorithms on medical image sets where our goal is to see how many different equivalence classes (i.e. non-redundant affinities) belong to a given image; in other words how big the search space is in the case of real applications.

2 Equivalence of affinities

Now, we will briefly present some standard concepts and definitions that will be used throughout, which are well known in fuzzy theory and more detailed descriptions can be found in the literature [10, 34, 35].

2.1 Basic notations and definitions

Let $\mathbb{Z}^n$ stand for the set of all n-tuples of integers. A binary fuzzy relation α on $\mathbb{Z}^n$ $(n \geq 2)$ is a *fuzzy adjacency* if α is symmetric and reflexive. The pair $\langle \mathbb{Z}^n, \alpha \rangle$ is called an n-dimensional *fuzzy digital space*. A *scene* over a fuzzy digital space $\langle \mathbb{Z}^n, \alpha \rangle$ is a pair $\mathcal{C} = \langle C, f \rangle$, where $C = \prod_{j=1}^{n} [-b_j, b_j] \subset \mathbb{Z}^n$, each $b_j > 0$ being an integer, and $f \colon C \to \mathbb{R}^k$ is a scene *intensity function* $(k \geq 1)$. If the range of f is a subset of the interval $[0, 1]$, the scene is called the *membership scene.*

2.2 Affinity functions

Affinity is a binary fuzzy relation which indicates how two spels hang together locally in the scene, its strength depending on how close these spels are spatially and how similar their properties are in the image. It plays a crucial role in the FC framework because the global fuzzy connectedness of spels is derived by means of their affinities. The following definition gives a general characterization of affinity functions [10].

Definition 1. *Let $\preceq$ be a linear order relation [13] on a set L and let C be an arbitrary finite non-empty set. A function $\kappa \colon C \times C \to \langle L, \preceq \rangle$ is an* affinity function *from C into $\langle L, \preceq \rangle$ if κ is symmetric and $\kappa(a, b) \preceq \kappa(c, c)$ for every $a, b, c \in C$.*

We say that κ is a *standard affinity* if it is a function taken from C to $\langle [0,1], \leq \rangle$. In practice, the value of $\kappa(c,d)$ depends on the adjacency strength $\alpha(c,d)$ of c and d and on the intensity function f. In our experiments, we use the following two well known types of affinitiestaken from the literature [11, 30].

Definition 2. *Let $\psi\colon C \times C \to \langle [0,1], \leq \rangle$ be a standard homogeneity-based affinity function such that for every $c, d \in C$*

$$\psi(c,d) = \begin{cases} 1 & \textit{if } c = d \\ e^{-|f(c)-f(d)|^2} & \textit{if } \|c-d\| = 1 \\ 0 & \textit{otherwise} \end{cases} \tag{1}$$

Definition 3. *Let $\phi\colon C \times C \to \langle [0,1], \leq \rangle$ denote a standard object feature-based affinity such that for every $c, d \in C$*

$$\phi(c,d) = \begin{cases} 1 & \textit{if } c = d \\ min(e^{-|f(c)-m|^2/\sigma^2}, e^{-|f(d)-m|^2/\sigma^2}) & \|c-d\| = 1 \\ 0 & \textit{otherwise} \end{cases}, \tag{2}$$

where m and σ are the expected mean and standard deviation values of object intensities.

We should remark that in Def. 3 a family of affinities are defined where affinities differ in parameter values m and σ (our expectations or a prior knowledge about the object). These values are usually associated with a given scene and therefore this association can be treated as an *affinity operator*

$$\mathcal{K}(\mathcal{C}, m, \sigma) := \langle \mathcal{C}, m, \sigma \rangle \mapsto \kappa_{m,\sigma},$$

which produces affinity functions based on its input parameters.

2.3 Fuzzy connectedness schemas

Next, we will briefly describe the concept of fuzzy connectedness and the algorithm schemas, but we will avoid any formal definitions and theorems because these can be found in the literature cited and are not too important in this study (due to the fact the concept of equivalent affinities is valid in all of these schemas).

Fuzzy connectedness is a binary fuzzy relation which refers to the global hanging togetherness of the spel pairs in a scene as follows. For a given spel pair c and d, we consider all possible connecting paths between them and the level of connectedness is defined by the maximum of the strengths of all paths. The strength of a path is the minimum of the affinities of consecutive spels along the path [34]. Intuitively, the higher the level of connectedness between to spels, the higher the probability that these spels belong to the same object.

There are some well-known and commonly used algorithm schemas (or FC frameworks) which are able to determine different segmentations for a given scene based on some affinity functions. These are absolute FC (AFC) [35], relative FC (RFC) [29], iterative RFC (IRFC) [12], scale-based [30] and vectorial FC [38].

2.4 Equivalent affinities

Equivalence of affinities [10, 11] is a key notion in our study. Informally, two affinity functions are equivalent in the FC sense if they lead to identical segmentations when applied to any scene starting from the same seeds. The following definition [10] characterizes this concept more formally, which constitutes the basis for our study here.

Definition 4. *The affinities* $\kappa_1 \colon C \times C \to \langle L_1, \preceq_1 \rangle$ *and* $\kappa_2 \colon C \times C \to \langle L_2, \preceq_2 \rangle$ *are equivalent in the FC sense if for every* $a, b, c, d \in C$

$$\kappa_1(a, b) \preceq_1 \kappa_1(c, d) \iff \kappa_2(a, b) \preceq_2 \kappa_2(c, d). \tag{3}$$

The statement (i.e. two equivalent affinities, as described in Def. 4 lead to identical segmentations), is presented and proven in [10] (Theorem 5).

Without loss of generality, we shall restrict our investigation to standard affinities due to the following theorem [10].

Theorem 1. *Every affinity function is equivalent in the FC sense to a standard affinity.*

Proof. See proof in [10]. □

3 General algorithm schema

One of the chief goals of our study is to provide algorithms that are able to determine equivalence classes for a set of affinities. This is important from a practical viewpoint because we should avoid the use of equivalent affinities during a real application. However, from a theoretical perspective, this provides the basis for investigating equivalence classes, like the number of classes compared to the cardinality of the affinity set. Here, we will present a general algorithm schema (mostly based on Def. 4), which is the first step towards defining these algorithms.

In our example, we will assume that we have an affinity operator which depends on a real parameter w, i.e. it provides a set of affinities. We should add that the operator may also depend on a given scene $\mathcal{C}$ and on some additional parameters, but we view these as fixed parameters due to the affinity equivalence being restricted to a particular scene. We propose an algorithm schema which takes an affinity operator and a scene as inputs and determines the set of affinity equivalence classes. It is an abstract algorithm because it is to be implemented for a particular family of affinity operators, highlighting the tasks and providing an algorithm template for more specific algorithms. We will present some actual applications later on.

More formally, assume that a fixed scene $\mathcal{C}$ is given and there is an affinity operator

$$\mathcal{K}(\mathcal{C}, w, p) := \langle \mathcal{C}, w, p \rangle \mapsto \kappa_w,$$

where w is the given parameter such that

$$w \in [L, U] \subseteq \mathbb{R}$$

and p represents all other parameters (which are dependent on $\mathcal{C}$). A certain affinity function for a given w is referred to as κ_w because the scene and the other parameters are fixed (hence the indices can be omitted to).

The following structure (defined in Def. 5) is the key component here because it provides a formal description of the set of affinity equivalence classes.

Definition 5. *Let γ be an equivalence relation on the interval $[L, U]$ such that*

$$\gamma = \{\langle w_1, w_2\rangle \in [L, U] \times [L, U] : \kappa_{w_1} \textit{ and } \kappa_{w_2} \textit{ are equivalent in FC sense}\},$$

and let G denote the set of the equivalence classes induced by γ.

The following definitions are used to construct the algorithm schema which determines G (Def. 5) based on the definition of equivalent affinities (Def. 4). First, suppose that a certain 4-tuple of spels $\langle a, b, c, d\rangle$ is fixed $(a, b, c, d \in C)$.

Definition 6. *Let $\Delta\colon [L, U] \to \{-1, 0, 1\}$ be a function such that*

$$\Delta(w) = sgn(\kappa_w(a, b) - \kappa_w(c, d)),$$

where sgn denotes the sign function.

Obviously, if $\Delta(w_1) = \Delta(w_2)$ then the corresponding κ_{w_1} and κ_{w_2} define the same ordering on the spel pairs (a, b) and (c, d).

Definition 7. *Let ρ be an equivalence relation on the interval $[L, U]$ such that*

$$\rho = \{\langle w_1, w_2\rangle \in [L, U] \times [L, U] : \Delta(w_1) = \Delta(w_2)\},$$

and let $P = \{P^{(-)}, P^{(0)}, P^{(+)}\}$ denote the set of the equivalence classes belonging to ρ, where

$$\begin{aligned} P^{(-)} &= \{w \in [L, U] : \Delta(w) = -1\},\\ P^{(0)} &= \{w \in [L, U] : \Delta(w) = 0\},\\ P^{(+)} &= \{w \in [L, U] : \Delta(w) = 1\}. \end{aligned}$$

Thus, each w in $P^{(-)}$ satisfies $\kappa_w(a, b) < \kappa_w(c, d)$. The sets G and P are partitions of $[L, U]$.

The general algorithm schema which computes G can be seen in Alg. 1. The procedure starts with an initial partition (step 1) which contains the interval $[L, U]$ itself. Then, it iterates over the possible 4-tuples (steps 2 - 9) and determines the set of equivalence classes P for each 4-tuple (Step 3). In the steps 4-8, the algorithm refines the current partition G with the elements of P, i.e. if P_{curr} and an element G_{curr} of G intersect, then the algorithm replaces G_{curr} by the intersection and the difference. The purpose of this step is to merge the different partitions of the 4-tuples into a global partition which describes γ and G (more formally, in Prop. 1). The performance of the partition refinement step depends on the general structure of the different partitions, and many algorithms and data structures used to implement this step can be found in the literature [24, 16, 27].

Algorithm 1 General algorithm schema

```
Input: the operator $\mathcal{K}$, the fixed $\mathcal{C}$, $p$ and the domain $[L, U] \subseteq \mathbb{R}$ of $w$
Output: $G$
 1: $G \leftarrow \{[L, U]\}$
 2: for all 4-tuple $\langle a, b, c, d\rangle$ do
 3:   determine $P = \{P^{(-)}, P^{(0)}, P^{(+)}\}$ (according to $\Delta$)
 4:   for all $P_{curr} \in P$ and $G_{curr} \in G$ do
 5:     if $P_{curr} \cap G_{curr} \neq \emptyset$ then
 6:       substitute $G_{curr}$ in $G$ by $G_{curr} \cap P_{curr}$ and $G_{curr} \setminus P_{curr}$
 7:     end if
 8:   end for
 9: end for
10: return $G$
```

Proposition 1. *The Alg. 1 computes G correctly.*

Proof. Suppose that the algorithm iterates over all possible 4-tuples in a given $t_1, t_2, \ldots, t_k$ order (where $t_i = \langle a, b, c, d\rangle$ is a 4-tuple of spels), and let P_i denote the partition P belonging to t_i and $G^{(i)}$ the state of G in the ith iteration. We would like to prove that in the ith iteration, $G^{(i)}$ consist of the equivalence classes belonging to the first i 4-tuples $(t_1, \ldots, t_i)$, i.e. γ is correct if its verification is restricted to these 4-tuples.

In the first step, when $i = 1$, the initial $[L, U]$ is partitioned by P_1, which means that $G^{(1)} = P_1$. Now, suppose that the statement is satisfied for i, thus γ is correct when restricted to $t_1, \ldots, t_i$. Then the method takes t_{i+1} and its partition $P_{i+1} = \{P^{(-)}_{i+1}, P^{(0)}_{i+1}, P^{(+)}_{i+1}\}$. Because P_{i+1} is a partition, each $G_{curr} \in G^{(i)}$ will be substituted by $P^{(-)}_{i+1} \cap G_{curr}$, $P^{(0)}_{i+1} \cap G_{curr}$ and $P^{(+)}_{i+1} \cap G_{curr}$, where at least one intersection is not empty.

Next, consider a non-empty intersection like $P^{(0)}_{i+1} \cap G_{curr}$, and let $w \in P^{(0)}_{i+1} \cap G_{curr}$. The affinities belonging to this set are equivalent in the FC sense restricted to $t_1, \ldots, t_i, t_{i+1}$, because for each parameter $w' \in P^{(0)}_{i+1} \cap G_{curr}$, the corresponding κ_w and $\kappa_{w'}$ define the same ordering on the spel pairs of the 4-tuples $t_1, \ldots, t_i, t_{i+1}$ due to the definition of G_{curr} $(t_1, \ldots, t_i)$ and due to the definition of $P^{(0)}_{i+1}$ (t_{i+1}). For each $w'' \in G_{curr} \setminus P^{(0)}_{i+1}$, κ_w and $\kappa_{w''}$ define a different ordering on t_{i+1}, and for each $w''' \in P^{(0)}_{i+1} \setminus G_{curr}$, κ_w and $\kappa_{w'''}$ define a different ordering at least once on the 4-tuples $t_1, \ldots, t_i$. Thus, $P^{(0)}_{i+1} \cap G_{curr}$ is an equivalence class in the FC sense restricted to $t_1, \ldots, t_i, t_{i+1}$. The proof is similar to $P^{(-)}_{i+1} \cap G_{curr}$ and $P^{(+)}_{i+1} \cap G_{curr}$. So the algorithm replaces all the subsets of $G^{(i)}$ by equivalence classes (restricted to the first $i + 1$ 4-tuples); and the induction step is satisfied. $\square$

4 Aggregating two affinities by weighted quasi-linear means

Next, we will investigate a more specific scenario, when a particular affinity combines two other affinities (e.g. homogeneity and object feature-based affinities) by means of an aggregation operator. These affinity functions are often used in real applications and as examples in the literature [7, 11, 23, 30, 34, 35]. Thus, in this example, our affinity operator depends on two affinity functions and an aggregation operator with a weight parameter w. The authors in [11] discuss the problem of combining affinities, and they use a weighted arithmetic mean, weighted geometric mean, and lexicographical order to aggregate affinity functions (other work on this topic can be found in [25]). Here, we study the first two, more precisely their general class i.e. quasi linear means, and we investigate the structure of equivalence classes and introduce several implementations of the general algorithm schema (Alg. 1) for this particular case.

4.1 The structure of equivalence classes regarding 4-tuple of spels

A characterization of quasi linear means can be found in Theorem 2 below [18]. We should add that this class of mean operators involves the weighted forms of arithmetic, geometric, harmonic and root-power means.

Theorem 2. *An operator $M^{(m)}$ is a continuous, strictly monotonic, idempotent and bisymmetrical if and only if $M^{(m)}$ represents a quasi-linear mean, i.e.*

$$M^{(m)}(x_1,\dots,x_m)=\varphi^{-1}\left(\sum_i \omega_i\varphi(x_i)\right),\quad \omega_i\geq 0,\ \sum\omega_i=1,$$

where $\varphi\colon[0,1]\to[0,1]$ is an increasing continuous function.

Proof. See [2, 1]. □

Definition 8. *Suppose that $\mathcal{C}=\langle C,f\rangle$ is a scene, $a,b\in C$, $\varphi\colon[0,1]\to[0,1]$ is a continuous increasing function, $\kappa_1,\kappa_2\colon C\times C\to\langle[0,1],\leq\rangle$ are standard affinity functions, and let $\mathcal{K}$ be an affinity operator such that*

$$\mathcal{K}(w,\mathcal{C},\kappa_1,\kappa_2):=\langle w,\mathcal{C},\kappa_1,\kappa_2\rangle\mapsto\kappa_w,$$

where $w\in[0,1]$ and $\kappa_w\colon C\times C\to\langle[0,1],\leq\rangle$ such that

$$\kappa_w(a,b)=\varphi^{-1}(w\cdot\varphi(\kappa_1(a,b))+(1-w)\cdot\varphi(\kappa_2(a,b))).$$

Clearly, κ_w is a weighted quasi-linear mean of the affinities κ_1 and κ_2 with the weights w and $1-w$, respectively.

Next, the function Δ defined in Def. 6 will have the following form (based on Def. 8):

$$\begin{aligned}\Delta(w) &= sgn(\kappa_w(a,b) - \kappa_w(c,d)) = \\ &= sgn(\varphi^{-1}(w \cdot \varphi(\kappa_1(a,b)) + (1-w)\cdot\varphi(\kappa_2(a,b))) \\ &\quad -\varphi^{-1}(w\cdot\varphi(\kappa_1(c,d)) + (1-w)\cdot\varphi(\kappa_2(c,d)))).\end{aligned}$$

Theorem 3 tells us that the partitions belonging to the 4-tuples have very simple structures in the case of quasi-linear means and this fact plays a crucial role when developing specialized algorithms.

Theorem 3. *Assume that $\langle a,b,c,d\rangle$ is a 4-tuple of spels ($a,b,c,d \in C$), and let κ_w be an affinity function, as defined in Def. 8. The partition P defined in Def. 7 (belonging to a,b,c,d) satisfies exactly one of the following statements:*
(1) $P = \{[0,1]\}$,
(2) $P = \{\{0\},(0,1]\}$ *or* $P = \{[0,1),\{1\}\}$,
(3) $P = \{[0,w^*),\{w^*\},(w^*,1]\}$ *for a* $w^* \in (0,1)$

Proof. Take the following constants:

$$X := \varphi(\kappa_1(a,b)), Y := \varphi(\kappa_2(a,b)), U := \varphi(\kappa_1(c,d)), V := \varphi(\kappa_2(c,d)).$$

In this case, Δ has the form:

$$\Delta(w) = sgn(\varphi^{-1}(w\cdot X + (1-w)\cdot Y) - \varphi^{-1}(w\cdot U + (1-w)\cdot V)).$$

Let l_1 and l_2 denote the following terms got from Δ:

$$\begin{aligned} l_1(w) &= w\cdot X + (1-w)\cdot Y &= w\cdot(X-Y)+Y, \\ l_2(w) &= w\cdot U + (1-w)\cdot V &= w\cdot(U-V)+V \end{aligned}$$

which are linear functions of w (actually they are two lines, if we interpret them on $\mathbb{R}$).
The functions φ and $\varphi^{-1}\colon [0,1]\to[0,1]$ are both bijections because they are invertible, and φ is increasing by definition; so φ and φ^{-1} are strictly increasing functions. Therefore the following hold for each $w \in [0,1]$:

$$\begin{array}{lllllllll} (L1) & l_1(w) < l_2(w) & \Rightarrow & \varphi^{-1}(l_1(w)) & < & \varphi^{-1}(l_2(w)) & \Rightarrow & \Delta(w) = & -1, \\ (L2) & l_1(w) = l_2(w) & \Rightarrow & \varphi^{-1}(l_1(w)) & = & \varphi^{-1}(l_2(w)) & \Rightarrow & \Delta(w) = & 0, \\ (L3) & l_1(w) > l_2(w) & \Rightarrow & \varphi^{-1}(l_1(w)) & > & \varphi^{-1}(l_2(w)) & \Rightarrow & \Delta(w) = & 1. \end{array}$$

In the following, we will show that the statements of the theorem can be derived from the relative position of the two lines l_1 and l_2 (which may be easily verified).

(a) If l_1 and l_2 are *identical* (i.e. $X = U$, $Y = V$), then $\Delta(w) = 0$ for each $w \in [0,1]$, hence $P = \{[0,1]\}$.

(b) If l_1 and l_2 are *not identical, but parallel*, i.e. $X - Y = U - V$ and $Y \neq V$, then $l_1(w) < l_2(w)$ or $l_1(w) > l_2(w)$ on whole $\mathbb{R}$, thus from $L1$ and $L3$, $\Delta(w) = -1$ or $\Delta(w) = 1$ for each $w \in [0,1]$, respectively. In this case, $P = \{[0,1]\}$ again.

(c) If l_1 and l_2 are *not parallel*, i.e. $X - Y \neq U - V$, so $X - Y - U + V \neq 0$, then they have an intersection in a given point $w^* \in (-\infty, \infty)$, which can be determine as follows:

$$w^* \cdot X + (1 - w^*) \cdot Y = w^* \cdot U + (1 - w^*) \cdot V,$$

and from here

$$\begin{array}{lcl} w^* \cdot X + (1 - w^*) \cdot Y & = & w^* \cdot U + (1 - w^*) \cdot V \\ w^* \cdot X + Y - w^* \cdot Y & = & w^* \cdot U + V - w^* \cdot V \\ w^* \cdot X - w^* \cdot Y + w^* \cdot V - w^* \cdot U & = & V - Y \end{array}$$

and finally, we can solve it for w^*:

$$w^* = \frac{V - Y}{X - Y + V - U}.$$

Because $X - Y - U + V \neq 0$, w^* is well-defined. There are three cases:

$$\begin{array}{lll} \textbf{(c.1)} \text{ If } w^* \notin [0,1] & \Rightarrow & P = \{[0,1]\} \\ \textbf{(c.2)} \text{ If } w^* \in \{0,1\} & \Rightarrow & P = \{\{0\}, (0,1]\} \text{ or } P = \{[0,1), \{1\}\}, \\ \textbf{(c.3)} \text{ If } w^* \in (0,1) & \Rightarrow & P = \{[0, w^*), \{w^*\}, (w^*, 1]\} \end{array}$$

In the case (c.1), $l_1(w) < l_2(w)$ or $l_1(w) > l_2(w)$ for each $w \in [0,1]$, so $\Delta(w) = -1$ or $\Delta(w) = 1$ are satisfied, as in the parallel case. Thus the whole $[0,1]$ constitutes one equivalence class. The case (c.2) differs from (c.1) in that Δ takes zero value in 0 or in 1, so there are two equivalence classes (the given endpoint of $[0,1]$ will be a class with one element). In (c.3), there are 3 classes: left from w^*, w^*, and right from w^* according to the relative position of the lines. □

4.2 Algorithms and their complexity

Based on Theorem 3, we can derive new algorithms from Alg. 1 which are specialized for the affinity operators defined in Def. 8.

Our first remark is that the partition refinement step by the interval $[0,1]$ is redundant, because each equivalence class will be replaced by itself (since $[0,1] \cap G_{curr} = G_{curr}$, $[0,1] \setminus G_{curr} = \emptyset$). Hence, if the cases (a), (b), (c.1) occur, the partition refinement step can be skipped. We consider that $[x,x) = (x,x] = \emptyset$ for each $x \in \mathbb{R}$. So we can treat the cases (c.2) and (c.3) together as e.g. $P = \{\{0\}, (0,1]\}$ is a special case of (c.3) when $w^* = 0$ and $P = \{[0,0) = \emptyset, \{0\}, (0,1]\} = \{\{0\}, (0,1]\}$. Notice that P is clearly defined by the dividing point w^*.

Following the previous statement, we can show that G can be described by $W = \langle w_1, w_2, \ldots, w_k \rangle$ which is the ascending ordered set of the dividing points $w^*_{(1)}, w^*_{(2)}, \ldots, w^*_{(k)}$ corresponding to the iterations of the Alg. 1 in which the cases (c.2) and (c.3) are satisfied. In the first iteration G is partitioned by P_1, so $G^{(1)} = \{[0, w^*_{(1)}), \{w^*_{(1)}\}, (w^*_{(1)}, 1]\}$. In the second iteration there are two cases. If $w^*_{(2)} = w^*_{(1)}$, then $G^{(2)} = G^{(1)}$. If $w^*_{(2)} \neq w^*_{(1)}$, then $w^*_{(2)}$ divides one of the intervals $[0, w^*_{(1)})$ and $(w^*_{(1)}, 1]$ into three parts. For example, let $w^*_{(2)} < w^*_{(1)}$. Then

$[0, w^*_{(1)})$ will be replaced by $[0, w^*_{(2)}), \{w^*_{(2)}\}, (w^*_{(2)}$ and $w^*_{(1)})$. Continuing this, we find that $G = \{[0, w_1), \{w_1\}, (w_1, w_2), \ldots, \{w_k\}, (w_k, 1]\}$, so we can define G by $W = \langle w_1, w_2, \ldots, w_k \rangle$.

The first algorithm specialized for the quasi-linear means can be found in Alg. 2. which was constructed based on our previous observations and Theorem 3. At the start, the set W is initialized. Then the method iterates over all of the possible 4-tuples of spels (steps 2-11). For a given 4-tuple, the constants X, Y, U, V are computed (steps 3-6). In Step 7, the algorithm checks to see whether the cases (a) or (b) occur (from Theorem 3), which means that any subsequent computations for that 4-tuple can be skipped (**continue** means that the iteration continues with the next 4-tuple). If the conditions are not satisfied, then the dividing point w^* is computed, and if W does not contain w^*, then W will be augmented by w^* (Step 10). Lastly, in Step 12, the algorithm orders the elements of W, and returns with the dividing points that represent the equivalence classes containing one element, and with the midpoints of the intervals between two dividing points; so it lists the class representatives of G.

Algorithm 2 Naive algorithm for quasi-linear means

```
Input: $\mathcal{C}$, $\kappa_1$, $\kappa_2$, $\varphi$
Output: the class representatives of $G$
1:  $W \leftarrow \emptyset$
2:  for all 4-tuple $\langle a, b, c, d \rangle$ do
3:      $X \leftarrow \varphi(\kappa_1(a, b))$
4:      $Y \leftarrow \varphi(\kappa_2(a, b))$
5:      $U \leftarrow \varphi(\kappa_1(c, d))$
6:      $V \leftarrow \varphi(\kappa_2(c, d))$
7:      if $X - Y - U + X = 0$ then continue
8:      $w^* \leftarrow \frac{V-Y}{X-Y+V-U}$
9:      if $w^* \notin [0, 1]$ then continue
10:     if $w^* \notin W$ then $W \leftarrow W \cup w^*$
11: end for
12: $\langle w_1, w_2, \ldots, w_k \rangle \leftarrow$ the ascending ordered set of $W$
13: return $\frac{0+w_1}{2}, w_1, \frac{w_1+w_2}{2}, w_2, \ldots, \frac{w_{k-1}+w_k}{2}, w_k, \frac{w_k+1}{2}$
```

We will now examine the complexity of Alg. 2. We will assume that W is a set implementation where the *add* and *contain* methods require a constant time (e.g. it is a hash set), and the algorithm performs an ordering on the elements of W in Step 12. The advantage of this approach is twofold: 1) if there are many repetitive elements, it costs less if we collect the different elements into an unordered set (with constant adding time) and then we have to sort fewer elements than maintaining an ordered set, 2) if we require just the number of the equivalence classes, we can omit the ordering step. Hence, in the following, we will omit Step 12 from our discussion and we will suppose that it is executed in $\mathcal{O}(|W| \cdot \log(|W|))$ or in $\mathcal{O}(|W|)$

time.

We view one iteration step (Step 3-10) as a constant time operation ($\mathcal{O}(1)$) because the computation of the values X, Y, U, V is always executed and it would be very difficult and time-consuming compared to the other operations (considering the constant time *add* method). Due to the above statements and considerations, the following holds.

Proposition 2. *Regardless the ordering of W, the time complexity of Alg. 2. is $\mathcal{O}(|C|^4)$.*

It is obvious that this complexity is unfeasible for real algorithms. In the following we propose two techniques which singificantly improve its performance.

First, we assume that each affinity function κ used by our framework satisfies the following. If the spels a and b are not neighbouring, then

$$\kappa(a, b) = 0.$$

Hence, it is sufficient if we consider only the neighbouring pixel pairs and avoid the redundant iterations, so we can modify Alg. 2. (see Alg. 3). The algorithm iterates over all possible pairs of neighbouring pixel pairs and computes w^* (Step 4) as in steps 3-10 in Alg. 2.

Algorithm 3 Algorithm for quasi-linear means - A

```
Input: $\mathcal{C}$, $\kappa_1$, $\kappa_2$, $\varphi$
Output: the class representatives of $G$
1: $W \leftarrow \emptyset$
2: for all neighbouring pixel pair $(a, b)$ do
3:     for all neighbouring pixel pair $(c, d)$ do
4:         compute $w^*$ for $\langle a, b, c, d\rangle$ and if it is valid then add it to $W$
5:     end for
6: end for
7: $\langle w_1, w_2, \ldots, w_k\rangle \leftarrow$ the ascending ordered set of $W$
8: return $\frac{0+w_1}{2}, w_1, \frac{w_1+w_2}{2}, w_2, \ldots, \frac{w_{k-1}+w_k}{2}, w_k, \frac{w_k+1}{2}$
```

Proposition 3. *Regardless the ordering of W, the time complexity of Alg. 3. is $\mathcal{O}(|C|^2)$.*

Proof. Suppose that each spel has a fixed number of neighbours denoted by k. Then the number of different neighbouring spel pairs is approximately $2k \cdot |C|$, i.e. $\mathcal{O}(|C|)$. Due to the nested for loops, the algorithm executes $\mathcal{O}(|C|^2)$ iterations. □

Note: For the sake of accuracy, if we repeatedly counted the spels which are not neighbours, the algorithm would execute a lot of redundant steps. Both for loops should contain a non-neighbouring pixel pair in order to cover this case exactly once.

Our last approach (Alg. 4.) extends the idea of Alg. 3. If the algorithm computes the same X, Y values (Alg. 2., steps 3-4) for the spel pairs (a_1, b_1), (a_2, b_2), then the pair (a_2, b_2) leads to a sequence of redundant iterations. Alg. 4. tries to avoid this kind of redundancy in such a way that it determines the set of different X, Y pairs for each neighbouring spel pairs (steps 2-7), and it again iterates over this set using two nested loops (steps 8-12).

Algorithm 4 Algorithm for quasi-linear means - B

```
Input: $\mathcal{C}$, $\kappa_1$, $\kappa_2$, $\varphi$
Output: the class representatives of $G$
 1: $W \leftarrow \emptyset$
 2: $S \leftarrow \emptyset$
 3: for all neighbouring pixel pair $(a, b)$ do
 4:     $X \leftarrow \varphi(\kappa_1(a, b))$
 5:     $Y \leftarrow \varphi(\kappa_2(a, b))$
 6:     if $(X, Y) \notin S$ then $S \leftarrow S \cup (X, Y)$
 7: end for
 8: for all $(X, Y) \in S$ do
 9:     for all $(U, V) \in S$ do
10:         compute $w^*$ for $\langle X, Y, U, V \rangle$ and if it is valid then add it to $W$
11:     end for
12: end for
13: $\langle w_1, w_2, \ldots, w_k \rangle \leftarrow$ the ascending ordered set of $W$
14: return $\frac{0+w_1}{2}, w_1, \frac{w_1+w_2}{2}, w_2, \ldots, \frac{w_{k-1}+w_k}{2}, w_k, \frac{w_k+1}{2}$
```

Proposition 4. *The time complexity of Alg. 4. regardless of the ordering of W is* $\mathcal{O}(|C| + |S|^2)$.

Proof. The determination of the set S (steps 3-7) requires $\mathcal{O}(|C|)$ time because one iteration step contains only a few constant time operations and the number of different neighbouring spel pairs is $\mathcal{O}(|C|)$, can be seen in Prop. 3. The nested loops in steps 8-12 require $\mathcal{O}(|S|^2)$ iterations, so the statement holds. □

Lastly, we should mention that we can make additional improvements by considering symmetries. E.g. $(a, b), (c, d) \equiv (c, d), (a, b)$ and $\kappa(a, b) = \kappa(b, a)$.

5 Experiments

Although our focus is mainly on theoretical results in this study (namely how we can characterize and determine the equivalence classes belonging to a certain type of affinity operators), we were also interested in testing the given algorithms on real images. Our aim here was to determine how many equivalence classes belong to a particular image and to measure the running times in practice. All the results shown in the following were measured on a PC with a 2 Ghz Intel Core i7 CPU and the algoritms were implemented in the Java programming language. In our experiments, two medical image sets were used : 1) 25 digital dermoscopy images of size 1280×1024 pixels, each contains one or more skin lesions, in RGB colour space (Fig. 1) and 2) 3×25 simulated brain MRI slices of size 181×217 (Fig. 2). Simulated T1, dual-echo T2, and proton density PD-weighted slices with 3% noise and 20% inhomogeneity were utilized [15, 14]. As base affinities (κ_1 and κ_2 in Def. 8), a standard homogeneity-based and a standard object feature-based functions were applied [10, 11]. We modelled 3 tests each for both datasets, and the results can be seen in Table 1. below. In each case, a membership scene was extracted according to the methodology of the particular domain, and then we applied Alg. 3 and Alg. 4 to determine the number of equivalence classes and measure the running times.

The results got from our test can be seen in Table 2. and Table 3. Along with the running times and number of iterations, the size of set S (Prop. 4) can be seen as well, which is the number of different (X, Y) pairs computed by Alg. 4. Case columns refer to the test cases defined in Table 1.

Testing on both datasets led to an enormous number of equivalence classes (about $10^6 - 10^7$). Alg. 3 is not feasible from a run time perspective, even on the smaller images (BrainWeb sets), while Alg. 4 needs just a few seconds as its improvements drastically reduced the required number of iterations, and Alg. 3 strongly depends on the size of image. The number of different (X, Y) pairs (Prop. 4) varies on different images, and does not reflect the image size.

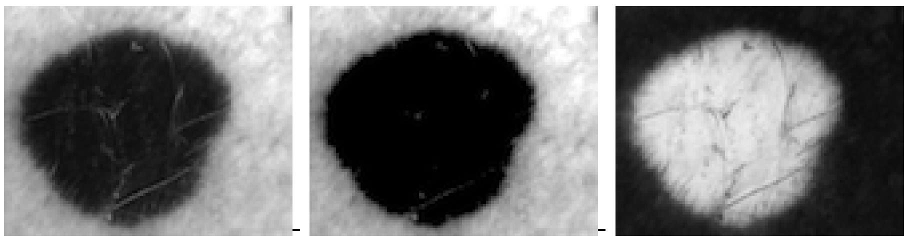
Figure 1: Dermoscopy images

6 Conclusions

The equivalence of affinities is a novel concept and it plays an important role in analyzing affinity functions in the FC framework. It tells us that we should note that different techniques used to defining affinity functions may lead to equivalent affinities, thus making these new constructions is unnecessary in a real application, as they only increase redundancy. Apart from the theoretical results, practical

[1]The blue channel in RGB colour space, proposed in [9, 8, 20, 17]

[2]A special membership scene in L*a*b* colour space where a given spel's membership value reflects to its colour distance from the average background colour [37]

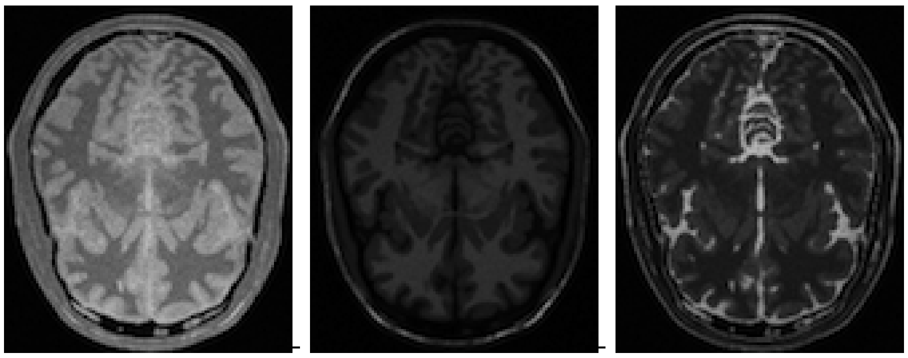

Figure 2: BrainWeb images

Dataset	#	Feature image(s)	Base affinities	Mean
Dermatology	1	greyscale	hom./obj.	geometric
	2	B channel [1]	hom./obj.	arithmetic
	3	a special scene [2]	hom./obj.	arithmetic
BrainWeb	1	T2 and PD	hom.	geometric
	2	T2	hom.	arithmetic
	3	T1	hom/obj.	geometric

Table 1: Test cases for the datasets. The membership scenes (feature images) are extracted according to the methodology of the particular domain. The expressions "hom." and "obj." stand for homogeneity-based and object feature-based affinities, respectively.

BrainWeb		Case-1	Case-2	Case-3
Number of parameters		2.13×10^7	6.16×10^6	1.74×10^6
Alg. 4	Running time	2.8 s	1.1 s	0.4 s
	Iterations	5.42×10^7	3.09×10^7	8.24×10^6
	(X, Y) pairs	10411	7857	4057
Alg. 3	Running time	420.3. s	451.9 s	438.0 s
	Iterations	1.22×10^{10}	1.22×10^{10}	1.22×10^{10}

Table 2: Results got on BrainWeb datasets. The expression "(X, Y) pairs" refers to the size of S in Prop. 4, which is an important factor in the time-complexity of Alg. 4.

Dermatology		Case-1	Case-2	Case-3
Number of parameters		4.51×10^6	1.71×10^7	3.48×10^7
Alg. 4	Running time	1.6 s	3.2 s	6.2 s
	Iterations	1.16×10^7	4.24×10^7	9.99×10^7
	(X, Y) pairs	4612	8909	14089
Alg. 3	Running time	≈ 141 h	≈ 141 h	≈ 141 h
	Iterations	1.37×10^{13}	1.37×10^{13}	1.37×10^{13}

Table 3: Results on dermatology images. The expression "(X, Y) pairs" refers to the size of S in Prop. 4 which is an important factor in the time-complexity of Alg. 4.

considerations can be derived as well. For instance, we could use integer arithmetic-based affinities in performance-sensitive applications.

In this paper, we focused on an example where the affinity operator has a parameter with a real value and it maps different affinity functions for different parameter values. These types of operators are used in a very common scenario when a homogeneity and an object feature-based affinity are combined. We constructed a general algorithm schema which could be a template for algorithms which are able to determine the equivalence classes of affinities according to a given affinity operator. Based on this template, we defined three algorithms for the example in which the above-mentioned affinities are combined using quasi-linear means. The complexity of these algorithms was also considered, and they were tested using two sets of medical images.

The structure of equivalence classes for quasi-linear means-based operators is quite simple and concise from a mathematical point of view. Furthermore, Alg. 4 required only a few seconds to process an image in our tests. Despite these points, the number of equivalence classes was enormous on the test images ($10^6 - 10^7$), which means we narrowed down the search space from the $[0, 1]$ interval[3] to a finite set of 10^7 elements. However, this value is still too high to explore all the different, non-equivalent affinities in a proper application, even if the experiments are performed in an automatic environment without human supervision.

There are many ways we could continue and improve the results of this study in the future. We did not analyze the relationship between the parameter values of the affinity operator and the corresponding segmentation results. We think that the reasonable number of different segmentations (and affinity functions) for a given image should be closer to $10 - 100$ than to 10^7, and the set of all non-redundant, non-equivalent affinities could be a good starting point to reduce this number. Other use cases could be also considered, when the parameter of affinity operator is not the weight of combination. Concrete algorithms built up from this schema will also give us information about complexity. The general algorithm schema could be extended to multiple variables and proper algorithms could be implemented.

[3]Obviously, in a proper implementation, we could use a floating point type which has a finite set of values

Acknowledgement

The author is grateful to all anonymous referees whose comments and suggestions have significantly improved our original version of this paper. The author is also grateful for the images provided by the Department of Dermatology and Allergology of Szeged. This study was partially supported by TÁMOP-4.2.2.C-11/1/KONV-2012-0005, Well-being in the Information Society.

References

[1] Aczél, J. On mean values. *Bull. Amer. Math. Soc*, 54(39):2–400, 1948.

[2] Aczél, J. and Dhombres, J.G. *Functional equations in several variables*, volume 31. Cambridge University Press, 1989.

[3] Beucher, S. et al. The watershed transformation applied to image segmentation. *SCANNING MICROSCOPY-SUPPLEMENT-*, pages 299–299, 1992.

[4] Boykov, Y., Veksler, O., and Zabih, R. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2001.

[5] Bustince, H., Barrenechea, E., and Pagola, M. Image thresholding using restricted equivalence functions and maximizing the measures of similarity. *Fuzzy Sets and Systems*, 158(5):496–516, 2007.

[6] Bustince, H., Barrenechea, E., and Pagola, M. Relationship between restricted dissimilarity functions, restricted equivalence functions and normal en-functions: Image thresholding invariant. *Pattern Recognition Letters*, 29(4):525–536, 2008.

[7] Carvalho, B.M., Gau, C.J., Herman, G.T., and Kong, T.Y. Algorithms for fuzzy segmentation. *Pattern Analysis & Applications*, 2(1):73–81, 1999.

[8] Celebi, M.E., Iyatomi, H., Schaefer, G., and Stoecker, W.V. Approximate lesion localization in dermoscopy images. *Skin Research and Technology*, 15(3):314–322, 2009.

[9] Celebi, M.E., Iyatomi, H., Schaefer, G., and Stoecker, W.V. Lesion border detection in dermoscopy images. *Computerized Medical Imaging and Graphics*, 33(2):148–153, 2009.

[10] Ciesielski, K.C. and Udupa, J.K. Affinity functions in fuzzy connectedness based image segmentation i: Equivalence of affinities. *Computer Vision and Image Understanding*, 114(1):146–154, 2010.

[11] Ciesielski, K.C. and Udupa, J.K. Affinity functions in fuzzy connectedness based image segmentation ii: Defining and recognizing truly novel affinities. *Computer Vision and Image Understanding*, 114(1):155–166, 2010.

[12] Ciesielski, K.C., Udupa, J.K., Saha, P.K., and Zhuge, Y. Iterative relative fuzzy connectedness for multiple objects with multiple seeds. *Computer Vision and Image Understanding*, 107(3):160–182, 2007.

[13] Ciesielski, Krzysztof. *Set theory for the working mathematician*, volume 39. Cambridge University Press, 1997.

[14] Cocosco, C.A., Kollokian, V., Kwan, R.K.S., Pike, G.B., and Evans, A.C. Brainweb: Online interface to a 3d MRI simulated brain database. In *NeuroImage*. Citeseer, 1997.

[15] Collins, D.L., Zijdenbos, A.P., Kollokian, V., Sled, J.G., Kabani, N.J., Holmes, C.J., and Evans, A.C. Design and construction of a realistic digital brain phantom. *Medical Imaging, IEEE Transactions on*, 17(3):463–468, 1998.

[16] Cormen, T.H., Leiserson, C.E., and Rivest, R.L. *Introduction to algorithms*. MIT Press, 2009.

[17] Emre Celebi, M., Alp Aslandogan, Y., Stoecker, W.V., Iyatomi, H., Oka, H., and Chen, X. Unsupervised border detection in dermoscopy images. *Skin Research and Technology*, 13(4):454–462, 2007.

[18] Fodor, J. and Roubens, M. *Fuzzy preference modelling and multicriteria decision support*, volume 14. Springer, 1994.

[19] Huang, L.K. and Wang, M.J.J. Image thresholding by minimizing the measures of fuzziness. *Pattern recognition*, 28(1):41–51, 1995.

[20] Iyatomi, H., Oka, H., Saito, M., Miyake, A., Kimoto, M., Yamagami, J., Kobayashi, S., Tanikawa, A., Hagiwara, M., Ogawa, K., et al. Quantitative assessment of tumour extraction from dermoscopy images and evaluation of computer-based extraction methods for an automatic melanoma diagnostic system. *Melanoma Research*, 16(2):183, 2006.

[21] Lei, T., Udupa, J.K., Saha, P.K., and Odhner, D. Artery-vein separation via MRA-an image processing approach. *Medical Imaging, IEEE Transactions on*, 20(8):689–703, 2001.

[22] Miki, Y., Grossman, R.I., Udupa, J.K., van Buchem, M.A., Wei, L., Phillips, M.D., Patel, U., McGowan, J.C., and Kolson, D.L. Differences between relapsing-remitting and chronic progressive multiple sclerosis as determined with quantitative MR imaging. *Radiology*, 210(3):769–774, 1999.

[23] Nyúl, L.G., Falcão, A.X., and Udupa, J.K. Fuzzy-connected 3d image segmentation at interactive speeds. *Graphical Models*, 64(5):259–281, 2002.

[24] Paige, R. and Tarjan, R.E. Three partition refinement algorithms. *SIAM Journal on Computing*, 16:973, 1987.

[25] Pednekar, Amol S and Kakadiaris, Ioannis A. Image segmentation based on fuzzy connectedness using dynamic weights. *Image Processing, IEEE Transactions on*, 15(6):1555–1562, 2006.

[26] Pham, D.L. and Prince, J.L. Adaptive fuzzy segmentation of magnetic resonance images. *Medical Imaging, IEEE Transactions on*, 18(9):737–752, 1999.

[27] Preparata, F.P. and Shamos, M.I. Computational geometry: an introduction, 1985. *New York.*

[28] Rice Jr, B.L. and Udupa, J.K. Clutter-free volume rendering for magnetic resonance angiography using fuzzy connectedness. *International Journal of Imaging Systems and Technology*, 11(1):62–70, 2000.

[29] Saha, P.K. and Udupa, J.K. Relative fuzzy connectedness among multiple objects: theory, algorithms, and applications in image segmentation. *Computer Vision and Image Understanding*, 82(1):42–56, 2001.

[30] Saha, P.K., Udupa, J.K., and Odhner, D. Scale-based fuzzy connected image segmentation: theory, algorithms, and validation. *Computer Vision and Image Understanding*, 77(2):145–174, 2000.

[31] Samarasekera, S., Udupa, J.K., Miki, Y., Wei, L., and Grossman, R.I. A new computer-assisted method for the quantification of enhancing lesions in multiple sclerosis. *Journal of computer assisted tomography*, 21(1):145–151, 1997.

[32] Sethian, J.A. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, volume 3. Cambridge university press, 1999.

[33] Tizhoosh, H.R. Image thresholding using type ii fuzzy sets. *Pattern recognition*, 38(12):2363–2372, 2005.

[34] Udupa, J.K. and Saha, P.K. Fuzzy connectedness and image segmentation. *Proceedings of the IEEE*, 91(10):1649–1669, 2003.

[35] Udupa, J.K. and Samarasekera, S. Fuzzy connectedness and object definition: theory, algorithms, and applications in image segmentation. *Graphical Models and Image Processing*, 58(3):246–261, 1996.

[36] Udupa, J.K., Wei, L., Samarasekera, S., Miki, Y., Van Buchem, MA, and Grossman, R.I. Multiple sclerosis lesion quantification using fuzzy-connectedness principles. *Medical Imaging, IEEE Transactions on*, 16(5):598–609, 1997.

[37] Xu, L., Jackowski, M., Goshtasby, A., Roseman, D., Bines, S., Yu, C., Dhawan, A., and Huntley, A. Segmentation of skin cancer images. *Image and Vision Computing*, 17(1):65–74, 1999.

[38] Zhuge, Y., Udupa, J.K., and Saha, P.K. Vectorial scale-based fuzzy-connected image segmentation. *Computer Vision and Image Understanding*, 101(3):177–193, 2006.