Multi-Seed Region Growing Algorithm for Medical Image Segmentation

Marco Gierlinger, Dinah Brandner, and Bernhard G. Zagar

Johannes Kepler University Linz, Institute for Measurement Technology, Altenberger Straße 69, AT-4040 Linz

Abstract We present a heuristic approach to segment an image into multiple regions for subsequent feature extraction. The algorithm is based on region growing and allows parallel implementation by employing multiple seeds, that independently grow a region until all pixels of the image have been assigned. Seeds are homogeneously dispersed in pixel space and the growth of regions is controlled by prioritizing neighboring pixels via a bucket queue. The heuristic is based on histograms that are built up during growth to derive binary images for each seed. These binary images are weighted by additive image fusion. A simple preprocessing technique is applied to tune the algorithm's outcome. We explain how input parameters influence the algorithm's outcome and how practical solutions can be obtained.

Keywords Image segmentation, region growing, feature extraction, image registration, parallel implementation

1 Introduction

In medical diagnostics, deep learning methods [1] allow for an increase in both sensitivity and specificity of diagnostic results. As a drawback, however, to obtain accurate results they rely on massive training data, which typically is not available in the required annotated quality since it requires labor-intensive labeling by experts. An unsupervised technique called region growing might improve that situation by providing fully automated computer-aided segmentation and feature extraction [2]. Region growing is used very extensively in medical diagnostic applications [3] and it has shown to be

DOI: 10.58895/ksp/1000124383-21 erschienen in:

Forum Bildverarbeitung 2020

DOI: 10.5445/KSP/1000124383 | https://www.ksp.kit.edu/site/books/m/10.58895/ksp/1000124383/

very useful, e. g. in the diagnosis of cardiac disease, or tumor volume segmentation [4]. It is an easy to implement and fast processing algorithm that is growing a region by comparing unassigned neighboring pixels to those already assigned to a growing region. It is, however, prone to so-called leakage. Without any special consideration or improvement to the algorithm, it tends to assign pixels also outside of a homogeneous region where borders are thinned out or interrupted due to noise or other artifacts. In this work, we address a noise-resistant and highly parallelizable technique, which can segment MRI volume data with a global view on the problem.

A further target is to design a tool for temporal analysis of image sequences by comparison of extracted features. A common issue in comparing two or more images is to register them, for example, at different instances of time, or when the sensor with respect to the patient is aligned differently. The task considered here tries to solve this issue by so-called image registration [5]. The idea of our work is to register different images by a set of extracted features based on region growing. However, this paper focuses on the region growing algorithm only and future work will cover the image registration part.

2 Related Work

Seeded Region Growing (SRG) by Adams and Bischof [6] is an effective and well-known image segmentation algorithm. SRG grows one or more regions, initially called seeds, that can be single-pixel-sized or a set of adjacent pixels. The algorithm grows these distinct regions due to some homogeneous criterion until all pixels are assigned a region. Formally, the s^{th} seed grows region A_s for every $s \in \mathbb{N}$ where s is less than or equal to a user-defined number of seeds k. Let $\vec{p} \in \mathbb{N}_0^3 = (p_0, p_1, p_2)$ be a pixel, where (p_0, p_1) is its position in pixel space and let $I(\vec{p}) = p_2$ be its intensity value. Let $N(\vec{p})$ be the set of neighboring pixels of $\vec{p}$ in pixel space and $N(A) = \{r \in A \mid N(r) \setminus A \neq \varnothing\}$ the neighboring pixels of region A. During an iteration, a pixel $\vec{p}$ is assigned to the region A_s if

$$\vec{p} \in N(A_s) \wedge \delta(A_s, \vec{p}) = \min_{\forall i \in \mathbb{N} \wedge i \leq k \ \forall \vec{y} \in N(A_i)} \{\delta(A_i, \vec{y})\}, \tag{2.1}$$

where function δ is defined as:

$$\delta(A, \vec{p}) = \left| I(\vec{p}) - \underset{\vec{y} \in A}{\text{mean}}\{I(\vec{y})\} \right| \tag{2.2}$$

The number of iterations equals the number of pixels, i. e. the algorithm halts when all pixels are partitioned. The condition of Eqn. (2.1) may hold for multiple regions A_s and a single pixel $\vec{p}$, however, according to the authors of the SRG implementation, a pixel cannot be assigned to multiple regions during a single iteration. As a consequence, two inherent order dependencies may occur, which may lead to different segmentation results as discussed by Mehnert and Jackway [7].

Anyway, the presented algorithm employs independent seeds, which can grow fully in parallel without a rendezvous before every pixel has been visited. Also, in this approach, the mean intensity of a region is neglected and growth is promoted where two directly neighboring pixels meet the condition of some homogeneous criterion only.

In many region growing algorithms, k seeds typically grow k regions, and selecting a proper set of seed positions is a non-trivial task and crucial to the outcome. In our approach, we use one or more seeds but positions are homogeneously dispersed in pixel space and the number of seeds does not necessarily equal the number of extracted regions.

3 Algorithm

The following algorithm is described for n-bit grayscale images in $\mathbb{N}_0^{w \times h}$, however, adapting it for higher dimensions should be straightforward. Multiple seeds are employed and aligned as grid with a user-defined width u and height v with $u, v \in \mathbb{N}$, $u \leq w$ and $v \leq h$. For every $k \in \mathbb{N}_0$, with $k < uv$, a seed position vector $(x, y) \in \mathbb{N}_0^2$ is defined as:

$$x = \left\lfloor \frac{w}{u}(k \mod u) + \frac{w}{2u} \right\rfloor, y = \left\lfloor \frac{h}{v} \left\lfloor \frac{k}{u} \right\rfloor + \frac{h}{2v} \right\rfloor, \tag{3.1}$$

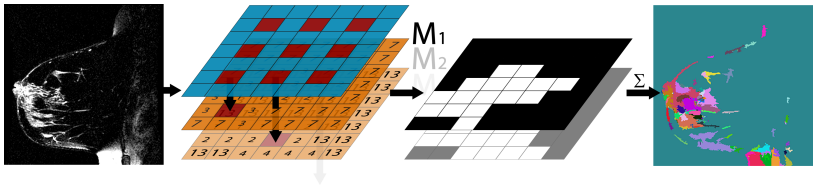

Figure 3.1: Region based segmentation applied to a breast MRI dataset [8].

as schematically depicted in red ($u = v = 3$) on the blue pane of Fig. 3.1. The algorithm walks through pixel space in an 8-adjacency flood-fill manner independently for each of the k seeds as follows: Let $\vec{p} \in \mathbb{N}_0^3 = (p_0, p_1, p_2)$ be a pixel, where (p_0, p_1) is its position in pixel space and let $I(\vec{p}) = p_2$ be its intensity value. A bucket queue is used to hold data objects $(\vec{p}, q)$, where $q \in \mathbb{N}_0$ with $q < 2^n$ is the bucket's index. Initially, the queue is filled with the seed only, which is a single pixel only. An iteration i is initiated by polling a bucket B_i. $\forall \vec{a} \in B_i \, \forall \vec{b} \in N'(\vec{a})$, a cost function $\delta_f : (\vec{a}, \vec{b}) \mapsto \{r \in \mathbb{N}_0 \mid r < 2^n\}$ is applied, where $N'(\vec{a})$ denotes all non-visited neighbors of $\vec{a}$. Cost function δ_f is defined as:

$$\delta_f(\vec{a}, \vec{b}) = \left| I(\vec{a}) - I(\vec{b}) \right|. \tag{3.2}$$

At the end of an iteration, each result is added to the queue as $(\vec{b}, \delta_f(\vec{a}, \vec{b}))$. A pixel counts as 'visited' when it is polled from (and not added to) the bucket queue.

3.1 Heuristic

Additionally, the number of newly assigned pixels m_i is tracked for each iteration i. A map $M_s \in \mathbb{N}_0^{w \times h}$, drawn as an orange pane in Fig. 3.1, is used for the s^{th} of k seeds and every $r_{a_0,a_1} \in M_s$, where (a_0, a_1) is the position of $\vec{a}$, is set to $m_{\max}(i) = \max(m_0, m_1, ..., m_i)$ at iteration i. Finally, when the queue is empty, i. e. every pixel was visited, M_s is converted into a binary map by

$$r \in M_s = \begin{cases} 0 & r < m_{\max}(i) \\ 1 & \text{otherwise} \end{cases} \tag{3.3}$$

Then, all k binary maps are added up elementwise to $\sum_{s=1}^{k} M_s$, i. e. additive image fusion is applied, and normalized to the range from zero to $2^n - 1$ to suppress regions that occur less often than others. An example is highlighted in Fig. 3.1 (right) with random colors assigned for regions containing the same numerical value. The following pseudo code gives an additional overview of the algorithm for a single seed:

Algorithm 1: Generating binary Map M_s for a single seed.

```
1  add pixel p⃗ = (x, y, 0) to queue;
2  set every r ∈ M_s and m_max to zero;
3  while queue is not empty do
4      poll bucket B (with highest priority) from queue;
5      set m_i to zero;
6      for each a⃗ ∈ B do
7          if a⃗ hasn't been visited yet then
8              mark a⃗ as visited;
9              increment m_i by one;
10             set value at position of a⃗ in Map M_s to m_max;
11             for each b⃗ ∈ N'(a⃗) do
12                 add (b⃗, f(a⃗, b⃗)) to queue ;
13     set m_max to max(m_max, m_i);
14 for each r ∈ M_s do
15     if r ≥ m_max then
16         set r to one;
```

In this algorithm, it is not necessary to visit every pixel. The iteration can halt when the number of non-visited pixels becomes smaller than $m_{\max}$. For the sake of simplicity, not every considered optimization is noted.

Growth is depicted in Fig. 3.2 and Fig. 3.3 for two arbitrary seed positions. Figure 3.2 shows a cumulative histogram of the pixel assimilation process and highlights the iteration for each seed where the largest peak is detected and Fig. 3.3 shows the corresponding growing process.

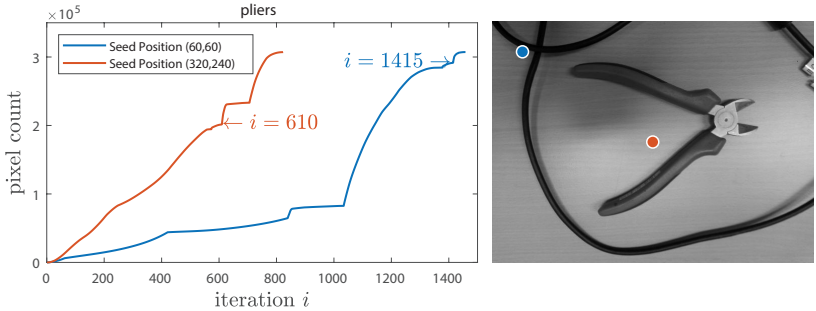

Figure 3.2: Left: Cumulative frequency of number of assigned pixels m_i on 'pliers' $(u, v) = (640, 480)$ and largest step $m_{\max}$ is found at highlighted iteration i. Right: Image with initial seed positions indicated.

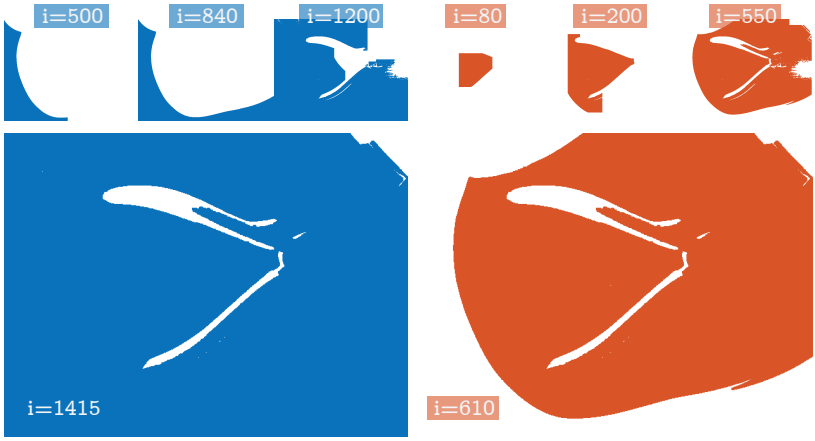

Figure 3.3: Growth of blue seed at $(60, 60)$ and orange seed at $(320, 240)$ on 'pliers'.

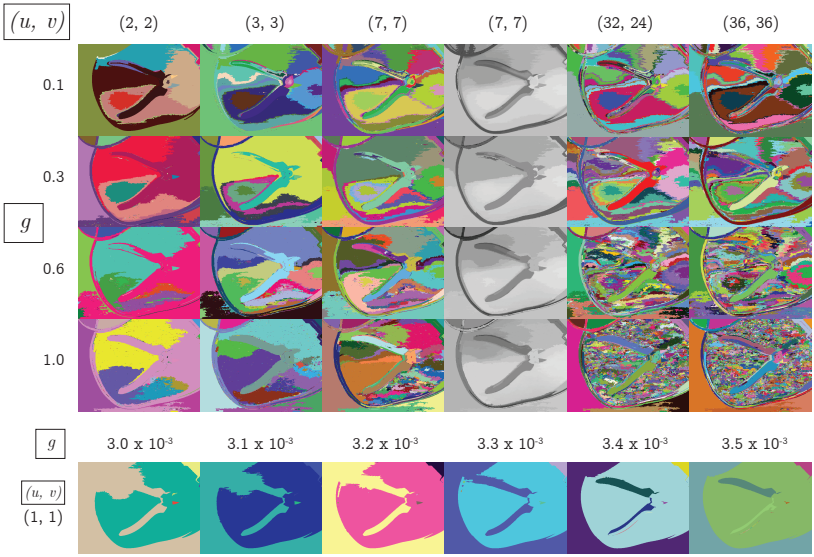

Figure 3.4: Tuning through the algorithm's solution space by adjusting the seed grid size uv and scaling parameter g. Random colors are assigned to distinctive regions. The fourth column from left contains the same regions as the third one and regions are colored by its mean intensity of the pixels of the original image.

3.2 Preprocessing

The algorithm's result is influenced by an input image and the selected grid size so far. However, it is desired to dynamically 'scan' for multiple acceptable solutions. Image quantization is used as a preprocessing step to decrease the image's bit depth, which has shown to be very useful to find practical solutions. Scaling parameter $g \in \mathbb{R}$ with $0 < g \leq 1$ is used to scale the image range to result in a lower bit depth. A user can tune the grid size uv, i. e. the density of dispersed seeds, and scaling parameter g as shown in Fig. 3.4 until a suitable solution is found. We may want to refer to [9], where image quantization is investigated when applied as a preprocessing step for dimensionality reduction in image classification pipelines.

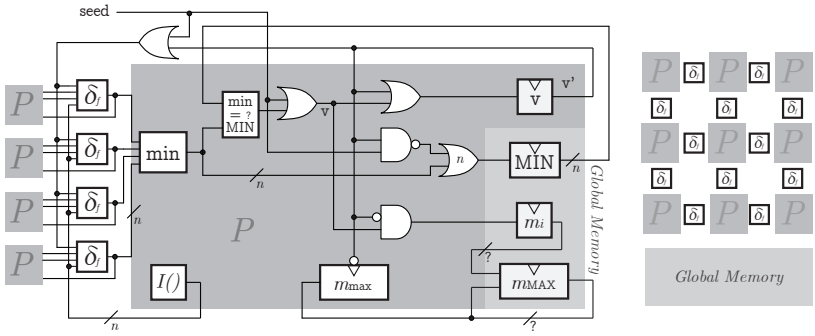

Figure 4.1: Sequential circuit representing a pixel in 4-adjacency. Left: Pixel in detail. Right: Pixel grid overview.

4 Parallel Implementation

This section intends to sketch the possibility of a highly parallelizable realization of the presented algorithm. While it is obvious to grow multiple seeds in parallel, it should also be noted, that per iteration, multiple pixels can potentially be examined in parallel as well. We present a concept for a sequential circuit, where the basic algorithm for the creation of a binary map is implemented such that it halts after as many clock cycles as iterations. The circuit is described for a pixel raster as schematically depicted on the right of Fig. 4.1, where $\boxed{\delta_f}$ denotes a connection in a 4-adjacent neighborhood. However, adapting it to 8-adjacency or 3D-connected cubes is straightforward. The single-bit input 'seed' (upper left of Fig. 4.1) is set HIGH for the seed pixel to initiate the algorithm and all essential blocks in Fig. 4.1 are implemented as:

- $\boxed{I()}$ outputs the pixel's intensity, which can be an unsigned integer between zero and $(2^n - 2)$, where n is the bit depth. The word $(2^n - 1)$ is defined as NULL within this section's context.
- v denotes single-bit memory, which saves the state whether a pixel was visited or not. It is set when the global MIN and

the local $\boxed{\text{min}}$ become equal and stays high until the algorithm halts.

- $\boxed{\delta_f}$ calculates δ_f of two neighboring pixels as in Eqn. (3.2), if the outputs $\boxed{\text{v}}$ of these two pixels differ from each other, i. e. one pixel is part of the region and one is a neighbor of it.
- $\boxed{\text{min}}$ propagates the minimum of the four n-bit neighboring $\boxed{\delta_f}$-results to the global $\boxed{\text{MIN}}$ if the pixel was already visited or if it is a seed, otherwise NULL is sent to $\boxed{\text{MIN}}$.
- $\boxed{\text{MIN}}$ finds the minimum of each's pixel $\boxed{\text{min}}$ output
- $\boxed{m_{\text{max}}}$ updates its value by the global $\boxed{m_{\text{MAX}}}$ output until the pixel was visited.
- $\boxed{m_i}$ is a parallel counter [10], which counts each pixel that is first-time visited. The result is compared to the previous value of $\boxed{m_{\text{MAX}}}$ to determine and output the maximum of both values.

The performance bottleneck consists of the $\boxed{\text{MIN}}$ and $\boxed{m_i}$ blocks, where a clever design is required to keep propagation delays low. However, propagation delay, stray capacitance, and any other hardware related issues are neglected in this section and require further investigation. The '?'-bit width should be chosen to count at least the largest number of bordering pixels that may occur at a single iteration.

5 Results and Discussions

5.1 Tuning, Merge and Split

As seen in Fig. 3.4, the more seeds are employed the more the algorithm tends to oversegmentation. The fewer seeds are employed, i. e. the smaller a seed grid size is selected, the more the position of a single seed influences the overall outcome of the algorithm. If only

a few homogeneous regions are dominating the image, increasing the number of seeds will not necessarily increase oversegmentation. However, the algorithm might be re-executed with already extracted regions as input instead of the whole image to furtherly split them. Conversely, oversegmentation can be compensated for by merging adjacent regions, that have a similar mean intensity.

This paper intends to provide a low-level tool for high-level applications. Whether the solution space has optimum solutions or not, is an application-dependent consideration and requires some sort of 'oracle' or 'teacher' as known from active learning [11].

5.2 Performance

When scaling parameter g is decreased the algorithm's time complexity decreases as well. While the bit depth and the spatial image resolution influence the algorithm's time complexity, it is believed that the complexity will not necessarily increase as the number of dimensions does for the parallel implementation. Analogously, one might compare the growth of a square in 2D, a cube in 3D, or a tesseract in 4D, where each dimension has the same spatial resolution.

5.3 Application and Future Work

When regions are extracted, subsequently, our goal is to register a large set of regions of MRI breast cancer image sequences. Also, we would like to apply our algorithm to pairs of stereo images like the Middlebury Stereo Datasets [12] to investigate the possibility of stereo matching techniques. Further investigations will cover non-rigid shape registration methods and similarity measure of how well registered regions match.

6 Conclusions

Based on seeded region growing, an algorithm was designed to support feature extraction in the field of medical diagnostics, however, it is not necessarily limited to this type of image. While the presented

algorithm is similar to the common region growing algorithms, the used heuristic is a novel and potentially faster approach. There is no need to find specific seed positions but instead, it is required to scale a parameter to adjust the density of homogeneously dispersed seeds in pixel space. Another input parameter is applied in a preprocessing step to reduce dimensionality by image quantization. The combination of adjusting density and dimensionality was depicted to give an intuition of the usefulness for more application-oriented approaches where optimal solutions might be found by an oracle as known from active learning.

A sequential circuit was described in this paper to point out the possibility of a highly parallel implementation with low time complexity. Overall experimental results seem promising, however, further investigation is required to evaluate the quality of segmentation.

Acknowledgement

This work has been supported by the LCM K2 Center within the framework of the Austrian COMET-K2 programme.

References

1. M. Ulrich, P. Follmann, J.-H. Neudeck, "A comparison of shape-based matching with deep-learning-based object detection," *tm – Technisches Messen, 2019, vol. 86(11), pp. 685–698*, 2019.
2. R. Neubecker, M. Heizmann, "Practice-oriented procedures for evaluating classifying image processing systems," *tm – Technisches Messen 2018, 85(4), pp. 252–267*, 2018.
3. Chowdhary, C.L., Acharjya, D.P., "Segmentation and feature extraction in medical imaging: A systematic review," *Procedia Computer Science, 167, pp. 26-36.*, 2020.
4. N. Mitschke, M. Heizmann, "About the detectability of objects in images by quantized neural networks," *tm – Technisches Messen, 2019, vol. 86(11), pp. 651–660*, 2019.
5. Z. Barbara, J. Flusser, "Image registration methods: a survey," *Image Vis. Comput., 21 (11) (2003), pp. 977-1000, 10.1016/s0262-8856(03)00137-9*, 2003.

6. R. Adams, L. Bischof, "Seeded region growing," *IEEE Transactions on Pattern Analysis and Machine Intelligence, 16 (6) (1994), pp. 641-647*, 1994.
7. A. Mehnert, P. Jackway, "An improved seeded region growing algorithm," *Pattern Recognit. Lett., 18 (1997), pp. 1065-1071, 10.1016/S0167-8655(97)00131-1*, 1997.
8. David Newitt, Nola Hylton, on behalf of the I-SPY 1 Network and ACRIN 6657 Trial Team, "Multi-center breast dce-mri data and segmentations from patients in the i-spy 1/acrin 6657 trials." *The Cancer Imaging Archive. http://doi.org/10.7937/K9/TCIA.2016.HdHpgJLK*, 2016.
9. Ponti M., Nazaré T.S., Thumé G.S., "Image quantization as a dimensionality reduction procedure in color and texture feature extraction," *Neurocomputing, 173 (2016), pp. 385-396*, 2016.
10. E. Swartzlander, "Parallel counters," *IEEE Transactions on Computers, vol. C-22, pp. 1021-1024*, 1973.
11. Burr Settles, "Active learning literature survey," *Computer Sciences Technical Report 1648, University of Wisconsin–Madison*, 2009.
12. D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nesic, X. Wang, and P. Westling, "High-resolution stereo datasets with subpixel-accurate ground truth," *German Conference on Pattern Recognition (GCPR 2014), Münster, Germany*, 2014.