Co-parent selection for fast region merging in pyramidal image segmentation

Milos Stojmenovic, Andres Solis-Montero, Amiya Nayak
[1] University of Ottawa, Canada
e-mails: mstoj075@site.uottawa.ca, amon@site.uottawa.ca, anayak@site.uottawa.ca

***Abstract*— The goal of image segmentation is to partition an image into regions that are internally homogeneous and heterogeneous with respect to other neighbouring regions. We build on the pyramid image segmentation work proposed by [3] and [9] by making a more efficient method by which children chose parents within the pyramid structure. Instead of considering only four immediate parents as in [3], in [9] each child node considers the neighbours of its candidate parent, and the candidate parents of its neighbouring nodes in the same level. In this paper, we also introduce the concept of a co-parent node for possible region merging at the end of each iteration. The new parents of the former children are co-parent candidates as if they are similar. The co-parent is chosen to be the one with the largest receptive field among candidate co-parents. Each child then additionally considers one more candidate, the co-parent of the previous parent. Other steps in the algorithm, and its overall layout, were also improved. The new algorithm is tested on a set of images. Our algorithm is fast (produces segmentations within seconds), results in the correct segmentation of elongated and large regions, very simple compared to plethora of existing algorithms, and appears competitive in segmentation quality with the best publicly available implementations. The major improvement over [9] is that it produces visually appealing results at earlier levels of pyramid segmentation, and not only at the top one.**

***Keywords*— image segmentation, pattern recognition.**

1. Introduction

Image segmentation can be very useful in many computer vision systems. It decomposes an image into homogeneous regions, which hopefully belong to the same object in the scene. Segmentation can be done according to some criteria by [7]. It is rarely achieved comprehensively for any single application, and algorithms that do perform well in one application are not suited for others.

Pyramid segmentation was proposed in [8, 10], and further elaborated on in [3]. Pyramids are hierarchical structures where each level is built by computing a set of local operations on the level below (with the original image being at base level or level 0 in the hierarchy). Level L consists of a matrix of points where each point contains some data and a link to at most one parent point in level L+1. The value of a point at levels higher than the base level is derived from the values of all its children points at the level below. When this is applied to children points transitively down to the base level, the value at each point at a given level is decided by the set of its descendent pixels at the base level (its receptive field). Each point at level L also represents an image component and constructs the image segmentation at level L-1, consisting of pixels belonging to its receptive field (if nonempty). Thus each level has its predefined maximum number of components. However its minimum number of components is left open and depends on a concrete image.

Image segmentation pyramids can be classified into regular and irregular types. Regular pyramids have a well-defined neighbourhood intra-level structure, where only natural neighbours in the mesh that defines a level of a pyramid are considered. Inter-level edges are the only relationships that can be changed to adapt the pyramid to the image layout. A constant reduction factor between levels is found in literature and pertains to regular pyramids [7]. Regular pyramids can suffer several problems [1, 7]: non-connectivity of the obtained receptive field, shift variance, or incapability to segment elongated objects.

To address the limitation of regular pyramids, irregular pyramids were proposed which vary in structure, increase the complexity and run time of the algorithms, and/or are dependant on prior knowledge of the image domain to be designed successfully. In the irregular pyramid framework, the relationships and reduction factors between levels are non-constant. Existing irregular pyramid based segmentation algorithms were surveyed in [7]. The described methods all appear very complex, have higher time complexities compared to regular pyramids and all consider the connectivity of the receptive field (base layer) as a design goal.

In [9], the authors observe that the connectivity of the receptive field (region in segmentation) is not always a desirable characteristic. For instance, in forestry and certain medical applications, the same type of vegetation or biomass could be present in several parts of the image, and treating them as a single segment may in fact be preferred for further processing. The proposed segmentation algorithm, called LS (Link Shifting), allows for non-connectivity of the receptive fields. Common receptive fields are coloured with common colours when the segmentation results are displayed.

The LS algorithm [9] is inspired by the regular pyramid linked approach (PLA) originally proposed by Burt et al. [3]. In the method proposed by [3], nodes from one level may alter their selection of parent at each iteration of the segmentation algorithm, which differs from previous pyramid structures. Starting from the base level (0), links between the current level L and level L+1 are decided. Each vertex at level L has four fixed candidate parents, and chooses the one which is the most similar from the higher level. The value of each parent is recalculated by averaging the values of its current children. Such iterations continue until the child-parent edges do not vary, or a certain number T of iterations is reached. The process then continues at the next higher level pair of levels. The main advantage of this method is that it does not need any threshold to compute the similarity between nodes at the same level. In the original method [3], each node must be linked to one parent node. Antonisse [1] introduced 'unforced linking' which allows the exclusion of some vertices from linking, and the presence of small components in the segmented image. Let *include(u)* be m times the standard deviation of a 3x3 neighbourhood around the node u (m is a parameter, whose value is 2 for 95% confidence and 3 for 99% confidence assuming a normal distribution of pixel intensities). This function is used to find the most similar parent q to node u. If u differs from q by at most *include(u), u* links to q. Otherwise, u fails to link to q and becomes the root of a new sub pyramid [1]. [9] changed this rule because 'unforced linking' of u and its parent was unnecessarily dependent on the neighbours of u in the existing rule from [1].

Antonisse [1] also proposed path fixing which defines some arbitrary path from the image(base) level to the top level to be fixed. This was indirectly applied here by the initial selection of random numbers for each pyramid vertex which were used for tie-breaking. Antonisse [1] also proposed randomized tie-breaking: when potential parent nodes have the same values, the choice of the parent is

978-1-4244-7249-9/10/$26.00 ©2010 IEEE

randomized. The tie-breaking rule has been completely redesigned in [9].

Burt et al [3] limit the choice of parent to 4 fixed nodes directly above the child. This approach has contributed to more successful segmentation, but due to the limited selection of parents, elongated and generally large segments that cover a significant portion of the image are not considered 'joined'.

We strive to design an algorithm that would properly handle elongated objects, while not enforcing connectivity of the receptive field and preserving shift invariance (the stability when minor shifts occur). This algorithm should also have favourable time complexity and execution time (within seconds, depending on the size of the images), and overall simplicity, so that it can be easily understood, implemented, and used in practice.

To achieve these goals, [9] made some simple changes in the way parent nodes are selected in the regular pyramid framework, which resulted in major improvements in their performance, including reduced shift variability and handling elongated objects. Instead of always comparing and selecting among the same four candidate parent nodes, each vertex at the current level selected the best node among its current parent, its current parent's neighbours, and the current parents of its neighbouring vertices at the same level. 4 connectivity neighbours were used in their implementation.

This paper makes numerous changes to the LS algorithm [9]. The new algorithm proposed here is called CPLS (Co-Parent Link Shifting). We describe bellow our new algorithm in full, and outline changes made from LS [9]. The main improvement was adding a co-parent as a candidate parent, which resulted in very fast region merging often at the first level of segmentation. This is due to the exponential progress of region merging in CPLS instead of linear progress in LS [9]. The parent selection rule was changed here from [3, 9]. In [3, 9], each pixel chooses the parent with the closest intensity to itself out of all candidate parents for its initial choice. The tie-breaking rule in [3] is as follows. When two or more 'distances' are equally minimal, one is chosen at random (randomization is done when needed). However, if one of them was selected as a parent in the previous iteration, it is then not changed. We observed a slow merging process when this rule is employed. For example, two neighbouring pixels would not change and merge two neighbouring parents when all of them have the same intensities. We have modified this rule. Enhancements include changing the region similarity test to eliminate fixed thresholds (the new test used here is based on statistical difference between regions, derived from means, deviations, and receptive field sizes), and a change in the rules for parent selection. If two parent candidates do not differ too much from the child node, the one with the larger receptive field is selected. If the receptive fields are the same size, the parent with the higher assigned random value is chosen. In [3, 9], the initial pixel values of each child are used to define the initial intensities of their parents. This was changed here, so that the final pixel intensities of children at level L are used to create the initial pixel intensities of their parents at level L+1. Here we consider a 5x5 candidate parent grid centered on the current parent after half of the levels of the pyramid have been traversed (algorithm LS [9] uses only 3x3 parent grid at each level). These changes resulted in major improvements in the algorithm's performance, while preserving its simplicity.

Experiments were conducted on both simple shapes, to validate the proposed methods, and on several real images. Evaluating segmentation quality in imagery is a subjective affair, and not easily done. "The ill-defined nature of the segmentation problem" [7] makes subjective judgment the generally adopted method (existing quantitative measurements are based on subjective formulas). In general, it is not clear what a 'good' segmentation is [7]. For this reason, no quantitative evaluation measure is applied to verify the results. The quality of the obtained segmentations appears visually satisfying for at least one level in each image. Our experiments compared the LS algorithm [9] with the new algorithm (CPLS) described here, and showed the superiority in the segmentation outcome.

2. Pyramid Image Segmentation Algorithm

Here we describe the proposed pyramid segmentation algorithm. The input to the algorithm is a greyscale, single channel image of dimensions 2^Nx2^N pixels, for $N\geq 2$. Each pixel (at level 0) u has integer value $I(u)$ in interval [0,255]. The output is the original image overlaid with the resultant segmentations at each level. In the pseudo code and discussion below, L is the level of the pyramid, L = 0, 1…, N. The bottom level (L = 0) is a matrix 2^N x 2^N pixels, representing the original image. Level L is matrix with 2^{N-L} rows and columns, i, j = 0, 1, 2 … 2^{N-L}-1. The top level $L=N$ has one element.

2.1 Creating the initial image pyramid at each level

We describe and use the overlapping image pyramid structure from [3]. Initially, the children of node $[i, j, L]$ are: $[i', j', L-1]=[2i+e, 2j+f, L-1]$, for $e,f \in \{-1,0,1,2\}$. There are a maximum of 16 children. That is, each $\{2i-1, 2i, 2i+1, 2i+2\}$ can be paired with each $\{2j-1, 2j, 2j+1, 2j+2\}$ to produce the coordinates of the 16 children. For $i=j=0$ there are 9 children, and for $i=0$ and $j>0$ there are 12 children. There are also maximum index values 2^{N-L+1}-1 for any child at level L-1 which also restricts the number of children close to the maximum row and column values. For $L=N$ there are four children of single node [0, 0, N] on the top: [0, 0, N-1], [0, 1, N-1], [1, 0, N-1], [1, 1, N-1] since the minimum index is 0 and the maximum is 1 at level N-1. Two neighbouring parents have overlapping initial children allocations. Conversely, each child $[i, j, L]$ for $L<N$ has 4 candidate parent nodes (if they exist) $[i'', j'', L+1]=[(i+e)/2, (j+f)/2, L+1]$, for $e,f\in\{-1,1\}$, where integer division is used (see Figure 1). Pixels at the edges of the image have fewer parents to choose from. The average intensity of all possible children is set as the initial intensity value $I(v)$ of the parent v, for nodes at levels $L>0$.

2.2 Testing similarity of two regions, unforced linking and the tie-breaking rule

Our algorithm makes use of a procedure for comparing the similarity of two regions, namely receptive fields of a vertex u and its candidate parent v. These two vertices are similar ($similar(u,v)=true$) if their intensities are roughly the same. In LS [9], a simple threshold $S=30$ for testing similarity was used. [9] also used a test for dissimilarity of two regions, to decide whether of not the best parent is an acceptable link, for the unforced linking option. A simple threshold based comparison against threshold value $D = 70$ was used in [9], and also here. The two functions are formally defined as follows.

Function *dissimilar(u,v)*
If $|I(u)-I(v)| > D$ **then** *dissimilar=true* **else** *dissimilar=false* .

Function *similar* is defined via a statistical test between receptive field distributions u and v when possible. It resulted in better image segmentations, but when an analogous improvement was attempted for the function *dissimilar* there was no further improvement, so only the simple version was used. In case receptive fields exist with single pixels, and therefore variance values of 0, the simple intensity difference test was used with $S = 15$. Let $n(u)$ denote the number of pixels in the receptive fields of u. Note that $n(u)$ is used in calculating $I(u)$ from intensity values of the children. For example, if w_1, w_2 and w_3 are children of u then $I(u) = (I(w_1)n(w_1)+I(w_2)n(w_2)+I(w_3)n(w_3))/n(u)$, $n(u) = n(w_1) + n(w_2) + n(w_3)$ (the intensity of the parent is the weighted sum of intensities of its children). Let $s(u)$ denote the variance of node u, that is, the variance of pixel intensities in its receptive field.

```
Similar (u, v) {
    If n(u) = 1 or n(v) = 1 {
        If |I(u)-I(v)| < S Then similar=true Else similar=false }
    Else {
    If n(u) <30 or n(v) <30
        $test = \frac{\left|I(u) - I(v)\right|}{\sqrt{\left(\frac{n(u)s(u)+n(v)s(v)}{n(u)+n(v)-2}\right)\left(\frac{1}{n(u)}+\frac{1}{n(v)}\right)}}$
    Else
        $test = \frac{\left|I(u) \ - I(v)\right|}{\sqrt{\left(\frac{s(u)}{n(u)}+\frac{s(v)}{n(v)}\right)}}$
    If test < 2 then similar=true Else similar=false  }
}
```

Each vertex *u* is initially assigned a random number *r(u)* in [0,1] which is never changed later on. Let *w* be a child node that compares parent candidates *u* and *v*, and let *better(w, u, v)* be one of *u* or *v* according to the comparison. The function is as follows.

```
Function better(w, u, v)
better=v;
If similar(w,u) and similar(w,v) then
    { if n(u)>n(v) or (n(u)=n(v) and r(u)>r(v)) then
better=u }
else if distance(w,u)<distance(w,v) or
    (distance(w,u)=distance(w,v) and n(u)>n(v)) or
    (distance(w,u)=distance(w,v) and n(u)=n(v) and
    r(u)>r(v))
then better=u;
```

In our current implementation, *distance(w, u)* = |*I(w)-I(u)*|. The function *better* normally selects the parent that is closer to the child node, based on the *distance* function, which is currently the difference in their pixel intensities. However, if they are both close (that is, *similar*) to the child node then the decision is made based on the size of their receptive fields, which is used as the secondary key in the comparison. If needed, the random numbers are used as ternary, tie breaking key for final arbitrage.

2.3 Candidate Parents

Among the candidate parents, each vertex at the level below selects the one which the closest to it, using the function *better* described above. For the first iteration, each vertex has up to four candidate parents, as per initial setup described by [3], and seen in Figure 1 (a). This fixed set of candidate parents has been changed (for further iterations) in our algorithm by a dynamic flexible set of candidate nodes that revolves around the current parent selection and the parent selection of neighbouring vertices at the same level. Suppose node $w = [i, j, L]$ is currently linked to parent $u = [i'', j'', L + 1]$ in iteration *t*, which we will denote simply by $p(w) = u$. The full notation would lead to $p[i, j, L][t]=[i'', j'', L + 1][t]$, and is convenient for easy listing of candidate parents. One set of candidate parents consists of the current parent and its 8 neighbours at the same level. Thus, in our notation, the candidate parents for the next iteration are: $[i''+ e, j''+ f, L + 1]$, where $e, f \in \{-1, 0, 1\}$. This produces a maximum of 9 candidate parents, which is a 3x3 grid centered at the currently linked parent. We also added four additional parent candidates, by considering the current selection (from the previous iteration) of neighbouring vertices at the same level. This is illustrated in Figure 1 (b). For $w=[i, j, L]$ and iteration $t+1$, we also consider $p[i+1, j, L][t]$, $p[i-1, j, L][t]$, $p[i, j+1, L][t]$, and $p[i, j-1, L][t]$ as parent candidates, if they exist. Both sets allow us to shift the parent further away in the next iteration, and possibly link the current child to a remote parent after the iterative process stabilizes (with no more changes in the selected parents). This change of parent selection nodes is directly responsible for the ability of our new algorithm to handle elongated objects. Note that we consider a 5x5 candidate parent grid centered on the current parent after half of the levels of the pyramid have been traversed by our algorithm. The segmentation quality was better than using 3x3 only or 5x5 only at all levels. This change does not however adversely impact the execution speed of the program since there are fewer children at higher levels of the pyramid, and raising the number of candidate parents from 13 to 29 does not constitute a significant increase in run time of the algorithm.

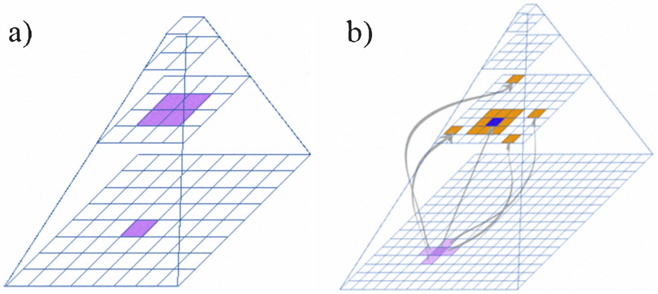


Figure 1 - Simple parent selection (a). 9 + 4 parent selection (b)

We introduce here a concept called 'co-parent' identification in candidate parent selection. Its main purpose is to unite similar segments early on in the segmentation algorithm. Similarly, the co-parent of parent *u*, at level *L+1*, denoted *c(u),* is a node at the same level as *u*, is similar to *u*, and at least one child at level *L* switched from *u* to *c(u)* at the end of a parent selection iteration. Even if several co-parent candidates are available, at most one co-parent is selected by picking the one with the largest receptive field (the random number is used to break the tie if needed). Let *p(w)* be the parent for node *w* at the end of the previous iteration, and let *p'(w)* be the selected (possibly new) parent of *w* after comparing 13 or 29 candidate parents. The co-parent of *u*, denoted *c(u)*, is calculated using the function *find-co-parent(u),* who's pseudo code is listed below.

```
Function find-co-parent(u)  //  returns c(u)
c(u)=-1                               // co-parent of u does not initially exist
For each child w at level L Do {  // p(w)=u
    If p'(w)   exists and p'(w)  ≠ p(w)   {
        If similar(u, p'(w)) {
            If c(u)=-1 or (similar(c(u), p'(w)) and
              ((n(p'(w))>n(c(u)) or (n(p'(w))=n(c(u)) and
              r(p'(w))>r(c(u))))
            then c(u)=p'(w) } }}
```

Once the co-parents of each parent are found at level *L*, each child tests its next iteration parent against the co-parent of its current parent. The better one of these two parents is selected as the next iteration parent.

2.4 Pyramid Segmentation

Once the pointers to parents have been initialized, the segmentation procedure may begin. Parent selections are attained at a given level (starting at level 0, and working toward the top of the pyramid) after a maximum of *T* iterations, before the process advances to the next level. At level *L*, each pixel initially points to the closest among four parents from the initial pyramid structure. In the subsequent iterations, it points to the (temporary) parent which best suits it in layer *L+1*, among the 9+4 or 25+4 candidate parents. This temporary parent is then compared to one more candidate, its co-parent, to yield the parent for the next iteration. The best parent is then tested for possible application of unforced linking based on dissimilarity. At the end of each iteration, the intensities of the parents in level *L+1* are recalculated based on the average intensity of the pixels in its current receptive field. Since these averages are calculated from the averages of its children, they must be appropriately weighted (by the number of pixels in the receptive fields of the children). Similarly the size of the receptive field, and the variance of the pixel intensities, are recalculated. Children that refused the link due to unforced linking (*unforced(w)=false*) are not considered in this calculation; however such children *w* continue looking for a parent in the next iteration. In case a child node has no current candidate parents, and its parent from the previous iteration has an empty receptive field, the child takes over that empty parent and transfers its receptive field onto it. This cycle of choosing parents, recalculating intensities, and reassigning parents continues for T = 10 iterations per pair of layers. The algorithm can be, at the top level, described as follows.

```
For levels $L$ = 0 to $N$-1 Do {
    For each parent node $u$ at level $L+1$ Do {
        Calculate initial parent intensity values, standard deviation,
        and receptive field size using 4x4 overlapping areas and
        default children. };
    For each child node $w$ in level $L$ Do {
                choose initial parent $p(w)$ among 4 default parents
                in level $L + 1$.
        If dissimilar(w, p(w)) Then unforced(w) = false
                                        Else unforced(w) = true. };
    For iter = 1 to $T$ Do {
        For each parent node $v$ at level $L+1$ Do {
            calculate new values for $I(v)$, $n(v)$, $s(v)$ based on
        children $u$ with
            $p(u)=v$ and unforced(u)=true };
        For each child $w$ at level $L$ Do {
            select parent $u$ among the 9+4 candidates (for $L < (N-1)/2$, and 25+4 otherwise) using method better (w, u,
            v) comparing currently best parent $u$ and a candidate $v$.
            $p'(w)= u$   /* temporary parent }
        For each parent $u$ at level $L+1$ Do {
            find co-parents $c(u)$ of u using function find-co-parent
            ($u$). }
        For each child $w$ at level $L$ Do {
            $p(w)$= better($w$, $c(p(w))$, $p'(w)$)  /* Compare $p'(w)$ with
            co-parent $c(p(w))$ of its parent $p(w)$ from the previous
            iteration to yield new parent $p(w)$ for the next iteration;
            If dissimilar(w, p(w)) Then unforced(w) = false
                                        Else unforced(w) = true. }
    }
}
Display segmentation for each level in pyramid.
```

3. Experimental Results

The algorithm presented here was designed to solve the problem of correctly segmenting elongated shapes in the framework of regular pyramid segmentation. It however works on various types of everyday imagery: both colour and greyscale, although colour images are converted to greyscale before processing begins. We have tested our algorithm on images of size 256 x 256 pixels. The processing time per image is 6 seconds for the images on a single core of a Pentium 1.66 GHz dual core machine, implemented in C# on the Windows XP operating system. Although it is relatively easy for us to judge whether or not a simple shape laid against a high contrast background is segmented properly, it is far more difficult to evaluate the precision of a segmentation of real imagery. The judgement of the quality of the segmentation is subjective.

3.1 Segmentation Results

The main benefit of the co-parent addition to the segmentation algorithm is that similar regions are joined much sooner in the pyramid segmentation structure. This principle is illustrated in Figure 2, where a simple shape on a white background is segmented by both algorithms. We see that the regions are much more quickly merged by the addition of co-parent consideration in the CPLS algorithm.

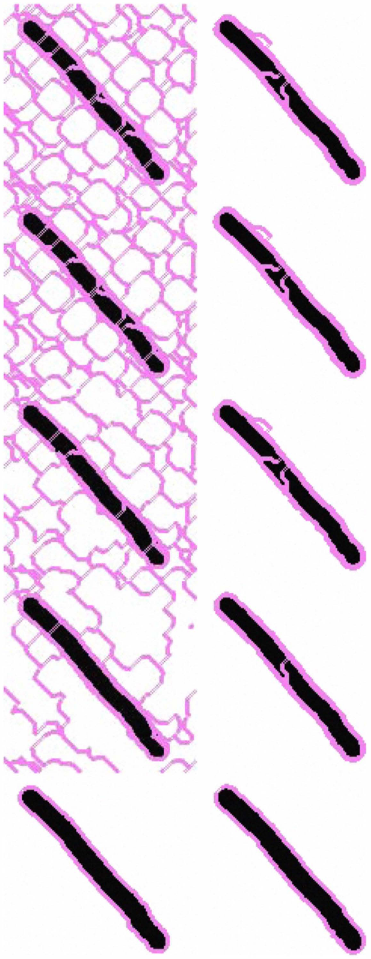


Figure 2 – LS (left), CPLS (right)

A second example of this effect is seen on a sample image in Figure 4 in the appendix. It shows an everyday image of a tree and the segmentation results for each level of the pyramid of the LS and CPLS algorithms.

Our algorithm was also applied to some examples of everyday imagery found in the Berkeley Segmentation Dataset [6]. Their images were cropped slightly so that they could be the correct size required by our algorithm. We compared our co-parent link shifting (CPLS) algorithm to the link shifting (LS) segmentation algorithm proposed by [9]. We also used the algorithm proposed by [2], who also employ a type of hierarchical segmentation structure but take into considerations texture as well as colour. The final algorithm used for comparison include the mean shift segmentation algorithm [5] implemented by [4] and named EDISON. All of the tested approaches, (including our own) are relatively parameterless, or the parameters have been set once, and remain consistent throughout testing for all of the tested images. In the case of [2, 4], their default settings were used in the implementations found. We have also fixed parameter values in our own implementation, as described in the text. The test results of all the algorithms are seen in Figure 3.

We show only the best level of segmentation of each pyramid since they best reflect the desired segmentation results for these images. The algorithms shown here have tendencies either to oversegment or undersegment images systematically, but sometimes perform adequately according to human observations. The algorithm of Christoudias et al. (CGM) [4] tends to oversegment images, and that of AGBB [2] tends to undersegment them. Our algorithm tends to oversegment areas that are textured, but that is to be expected since it is only pixel intensity based, and does not have any concept of texture. In the cases of coarsely textured images, AGBB [2] performs best. However, based on this selection of images from [6], the algorithms are fairly competitive. The CPLS algorithm better and more efficiently grouped similar regions at lower levels of segmentation, which is what it was designed to do.

4. Future Work

Since our CPLS method relies on greyscale intensities, it has a tendency of over-segmenting textured regions that appear homogenous to human observers, yet distinguishably heterogeneous to the program. One general problem with our approach is the lack of consideration of texture in segmentation. This is best illustrated in the first example in Figure 3. The airplane is segmented correctly since it is homogeneous in colour, yet the clouds in the background are not understood to belong to the same segment since they demonstrate slight variability in colour. The selection of the best level of segmentation appears possible given a set of constraints or requirements, but is not the same for every picture, and eventually a human observer may be needed to select the best outcome for each picture, according to its further processing needs.

The algorithm can be modified in a variety of ways. We have tested a variety of options for functions *dissimilar* and *distance*, involving standard deviations, but none of them improved the outcome. However, there are other possible definitions for these functions that could be tested.

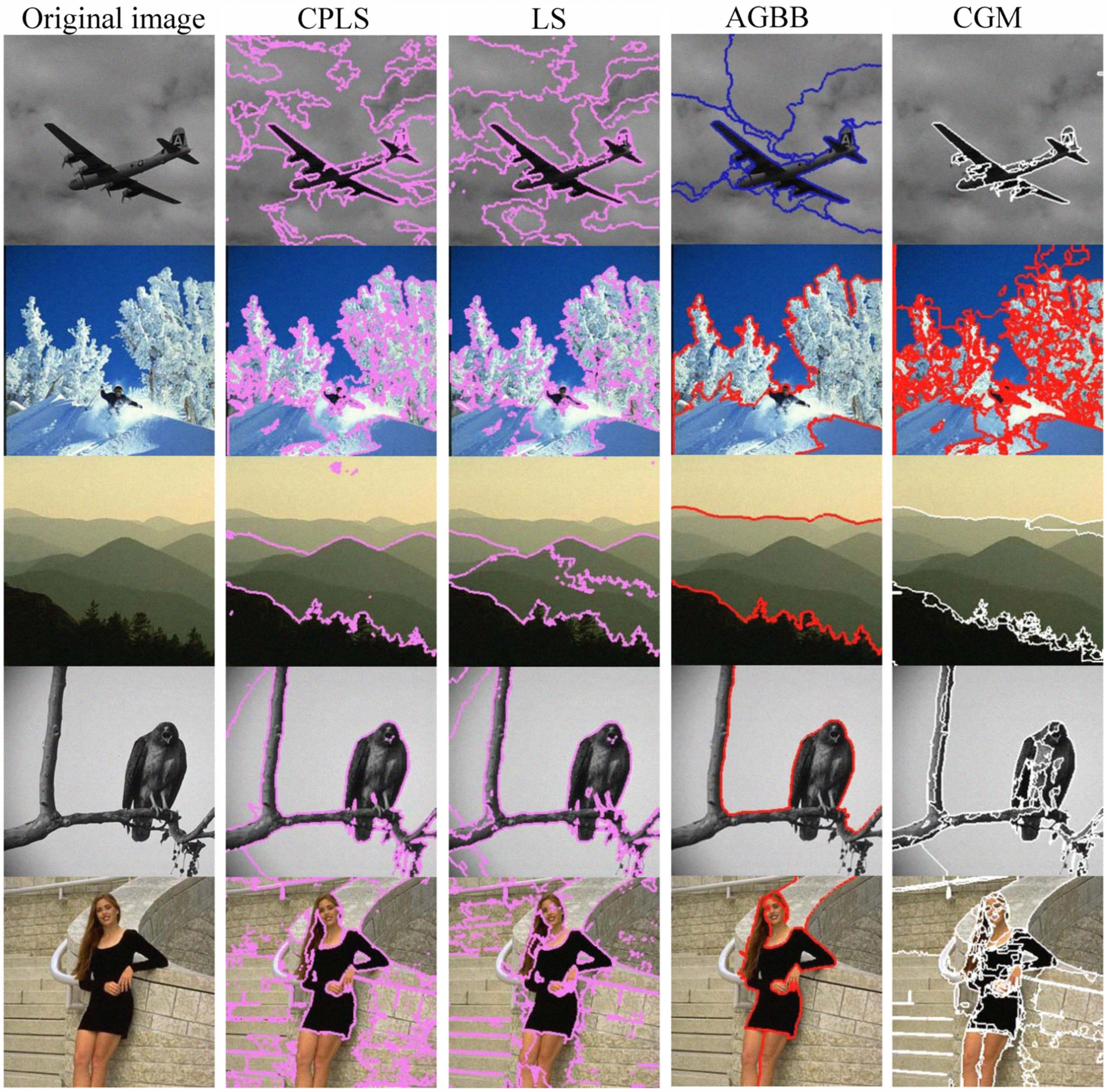


Figure 3 – Sample images and segmentations

The algorithm can also be modified to enforce connectivity of receptive fields, either at the very end (applying a connected components algorithm to subdivide a region into connected pieces), or similarly splitting parents during the parent selection process.

To improve the outcome of this segmentation algorithm, one would have to have at least some prior knowledge of the scene that is to be segmented. Such knowledge includes the minimum possible segment size, and possibly a range of pixel intensities within a region that could be considered homogenous. Other solutions may include considering more than just greyscale intensities of input data. In the current implementation, just the RGB layers are considered, and they are combined into just a single layer greyscale representation of the original image. By considering the Euclidean distance between two 3D points in an RGB space instead of simply considering greyscale differences, more accurate parent selection could be achieved at the expense of increased computation time. Furthermore, the current structure can be manipulated to consider pictures that are not only of size $2^n x 2^n$, but instead any size, including rectangular shapes. This size restriction was important in the original implementation of [3], but due to the modification of the parent selection process, it may be possible to eliminate this criterion.

We have used and experimented with the overlapping image pyramid structure as originally proposed in [3]. This refers to the fact that parent vertices at level 1 have overlapping receptive fields. Antonisse [1] already argued that perhaps a non-overlapping structure could perform better. We left this modification for further study, so that we can first investigate the impact of a single major change proposed here, the use of flexible parent links.

In this research, we attempted to find the best choices of parameters that will suit all images without adjusting them when a particular image is segmented. It is obviously possible to improve segmentation quality by manually adjusting parameter values and redoing segmentation until a plausible outcome is obtained for each picture. This would however defeat the purpose of an automated segmentation algorithm, and reduce it to human assisted segmentation. In our future research, we will also consider automatic selection of best parameter values for any given image. This can be achieved by using an independent measure of image segmentation quality.

5. References

[1] H. Antonisse, Image Segmentation in Pyramids, *Computer Graphics & Image Processing,* 19, 367-383, 1982.

[2] S. Alpert, M. Galun, R. Basri, A. Brandt, Image segmentation by probabilistic bottom-up aggregation and cue integration, *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR-07),* 2007.

[3] P. Burt, T. Hong, A. Rosenfeld, Segmentation and estimation of image region properties through cooperative hierarchical computation, *IEEE Trans. Systems, Man, and Cybernetics,* Vol. 11, No. 12, 1981.

[4] C. Christoudias, B. Georgescu, and P. Meer, Synergism in low level vision, *Proc. Int'l Conf. Pattern Recognition,* vol. 4, pp. 150-156, 2002.

[5] D. Comaniciu and P. Meer, Mean shift: A robust approach toward feature space analysis, *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, pp. 603-619, 2002.

[6] D. Martin and C. Fowlkes and D. Tal and J. Malik, A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics, Proc. 8th Int'l Conf. Computer Vision (ICCV), Vol. 2, pp. 416-423, July 2001.

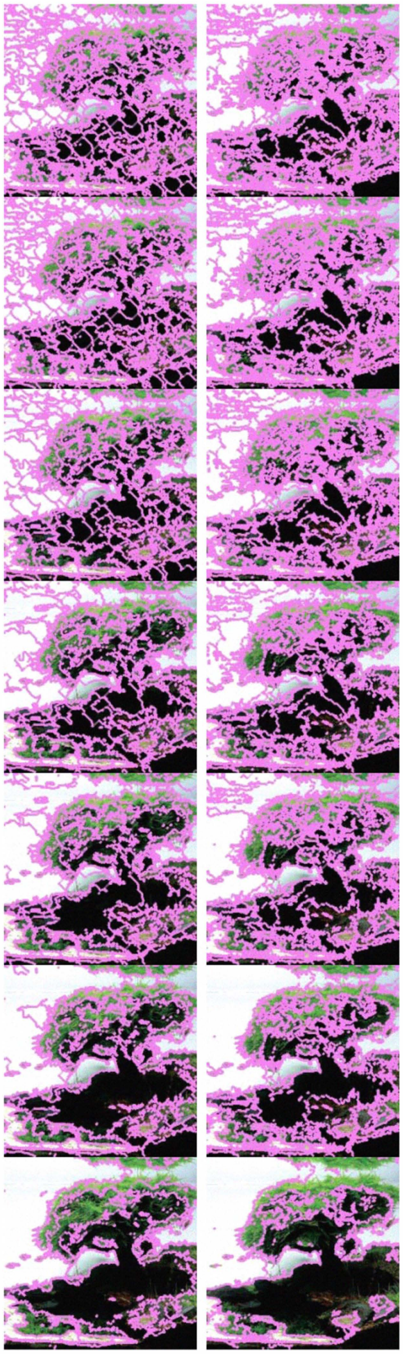
[7] R. Marfil, L. Molina-Tanco, A. Bandera, J. Rodriguez, F. Sandoval, Pyramid segmentation algorithms revisited, *Pattern Recognition,* Vol. 39, pp. 1430-1451, 2006.

[8] E. Riseman, A. Hanson, Design of a semantically-directed vision processor, Tech. Rep. 74C-1, Department of Computer Information Science, University of Massachusetts, Amherst, MA, 1974.

[9] M. Stojmenovic, A. Solis-Montero, A. Nayak, Link shifting based pyramid segmentation for elongated regions, *Int. Symposium on Signal Processing, Image Processing and Pattern Recognition SIP 2009,* Jeju Island, Korea, December 10~12, 2009, CCIS 61 (D. Slezak et al. eds.), Springer-Verlag, 141-152, 2009.

[10] S. Tanimoto, T. Pavlidis, A hierarchical data structure for picture processing, Computer Graphics and Image Processing, Vol. 4, pp. 104-119, 1975.

5. Appendix



Figrue 4 – LS (left) and CPLS (right) segmentation pyramid results