Tuning Range Image Segmentation by Genetic Algorithm

Gianluca Pignalberi
Dipartimento di Informatica, Università di Roma La Sapienza,
Via Salaria, 113 00198 Roma, Italy.
E-mail: pignalbe@dsi.uniroma1.it

Rita Cucchiara
Dipartimento di Ingegneria dell'Informazione, Università di Modena e Reggio Emilia,
Via Vignolese, 905 41100 Modena, Italy.
E-mail: rita.cucchiara@unimo.it

Luigi Cinque
Dipartimento di Informatica, Università di Roma La Sapienza,
Via Salaria, 113 00198 Roma, Italy.
E-mail: cinque@dsi.uniroma1.it

Stefano Levialdi
Dipartimento di Informatica, Università di Roma La Sapienza,
Via Salaria, 113 00198 Roma, Italy.
E-mail: levialdi@dsi.uniroma1.it

***Abstract*— Several range image segmentation algorithms have been proposed, each one to be tuned by a number of parameters in order to provide accurate results on a given class of image. Segmentation parameters are generally affected by the type of surfaces (e.g. planar vs. curved) and the nature of the acquisition system (e.g. laser range finders or structured light scanners).**

It is impossible to answer the question "which is the best set of parameters given a range image within a class and a range segmentation algorithm?". Systems proposing such a parameter optimization are often based either on careful selection or solution space-partitioning methods. Their main drawback is that they have to limit their search to a subset of the solution space to provide an answer in acceptable time. In order to provide a different automated method to search a larger solution space, and possibly to answer more effectively the above question, we propose a tuning system based on genetic algorithms.

A complete set of tests was performed over a range of different images and with different segmentation algorithms. Our system provided a particularly high degree of effectiveness, in terms of segmentation quality and search time.

***Index Terms*—Range images, segmentation, genetic algorithms.**

I. Introduction

Image segmentation problems can be approached with several solution methods. The range image segmentation sub-field has been addressed in different ways. But, since an algorithm should work correctly for a large number of images in a class, such a program is normally characterized by a high number of tuning parameters in order to obtain a correct, or at least satisfactory, segmentation.

Usually the correct set of parameters is given by the developers of the segmentation algorithm, and it is expected to give satisfactory segmentations for the images in the class used to tune the parameters. But it is possible that, given changing input image class, the results are not satisfactory. To avoid exhaustive-test tuning, an expert system to tune parameters should be proposed. In this way should be possible to easily direct the chosen segmentation algorithm to work correctly with a chosen class of images.

Several expert systems have been proposed by other teams. We can quote [1], that performs the tuning of a color image segmentation algorithm by a genetic algorithm. The same technique can be applied to range segmentation algorithms. Up to now, only techniques that partition the parameter space and work on a successive approximation have been used (such as in [2], [3], [4], [5]). Such techniques obtain results similar to those provided by the algorithm teams' tuning.

In this paper we propose a tuning system based on genetic algorithms. To prove the validity of this method we will show results obtained using well-tuned segmentation algorithms of range images (in particular the ones proposed at the University of Bern and University of South Florida). Genetic solutions are evaluated according to a fitness function that accounts for different types of errors, such as under/over-segmentation or miss-segmentation.

The paper is organized as follows. In section II we summarize the related works. In section III we describe in detail our approach. In section IV we show the experimental results, while in section V we present our conclusions.

II. Related works

A. Range image segmentation

Range images are colored according to the distance from the sensor that scans the image. In fact each pixel in a range image indicates the value of the distance from the sensor to the foreground object point. Image segmentation is the refinement of an image into patches corresponding to the represented regions. So the range image segmentation algorithm aims at partitioning and labeling range images into surface patches, that correspond to surfaces of 3D objects.

Surface segmentation is still a challenging problem. Currently many different approaches have been proposed. The known algorithms devoted to range segmentation may be subdivided into at least three broad categories [6]:

1) those based on a region-growing strategy;
2) based on clustering method;
3) based on edge detection and completion followed by surface filling.

Many algorithms addressing range segmentation have been proposed. In [6] there is a complete analysis of four segmentation algorithms –from the University of South Florida (USF), the University of Bern (UB), the Washington State University (WSU) and the University of Edinburgh (UE)–. The authors show that a careful parameter tuning has to be performed according to the chosen segmentation algorithm and image set. Such algorithms are based on the above methods, and show different performances and results in terms of segmentation quality and segmentation time.

Jiang and Bunke [7] describe an evolution of the segmentation algorithm built at the University of Bern and in [5] the same segmentation algorithm is used for other tests. Recently a different segmentation algorithm was presented, based on the scan-line grouping technique [8], but using a region-growing strategy, and showing good segmentation results and a quasi-real-time computation capability. Zhang et al. [9] presented two algorithms, both edge-based, segmenting noisy range images. By these algorithms the authors investigated the use of the intensity edge maps (IEMs) in noisy range image segmentation, and the results compared against the corresponding obtained without using IEMs. Such algorithms use watershed and scan-line grouping techniques. Chang and Park [10] proposed a segmentation of range images based on the fusion of range and intensity images, and the estimation of parameters for surface patches representation is performed by a "least-trimmed-squares" (LTS) method. Baccar et al. [11] describe a method to extract, via classification, edges from noisy range images. Several algorithms (particularly color segmentation algorithms) are described or summarized in [12].

Parameters tuning is still a main task and a possible solution is proposed. A different method to tune set parameters is given by Min et al. in [2], [3], [4]. The main drawback seems to be that a limited subset of the complete solution space is allowed to be explored, but exposes the method to the possibility of missing the global optimum or a good enough local optimum. But such a method is fast and efficient enough to represent a fine-tuning step: given a set of rough local sub-optima, the algorithm proposed in [2] could quickly explore a limited space around these sub-optima, to reach, if they exist, local optima.

In [6], for the first time, an objective performance comparison of range segmentation algorithms has been proposed. Further results on such a comparison have been proposed in [13], [14], [3], [4]. Another comparison has been presented in [15], where another range segmentation algorithm is proposed. This is based on a robust clustering method (used also for other tasks). But the need for tuning algorithm parameters is still present.

B. Genetic algorithms and their application to image segmentation

Genetic Algorithm (GA from now) is a well known spread technique for exploring in parallel a solution space by encoding the concept of evolution in the algorithmic search: from a *population* of individuals representing possible problem solutions, evolution is carried out by means of selection and reproduction of new solutions. Basic principles of GAs are now well known. Quoted references are the books of Goldberg [16] and Michalewicz [17]; a survey is presented in [18], while a detailed explanation of a basic GA for solving NP-hard optimization problem, presented by Bhanu et al., can be found in [1].

Many GA driven segmentation algorithms have been proposed in the literature; in particular, an interesting solution was presented by Yu et al. [19]: an algorithm that can segment and reconstruct range images via a method called RESC (RESidual Consensus). Chun and Yang [20] presented an intensity images segmentation by a GA addressed split-and-merge and Andrey and Tarroux [21] proposed an algorithm which can segment intensity images by including production rules in the *chromosome*, i.e. a data string representing all the possible features present in a population member. Methods for segmenting textured images are described by Yoshimura and Oe [22] and Tsang and Lai [23]. The first one adopts a "small region"-representing chromosome, while the second one uses GAs to improve the iterated conditional modes (ICM) algorithm [24]. Cagnoni et al. [25] presented a GA based on a small set of manually-traced contours of the structure of interest (anatomical structures in three-dimensional medical images). The method combines the good trade-off between simplicity and versatility offered by polynomial filters with the regularization properties that characterize elastic-contour models. Andrey [26] proposed another interesting work, in which the image to be segmented is considered as an artificial environment. In it, regions with different characteristics are presented as a set of ecological niches. A GA is then used to evolve a population distributed all over this environment. The GA-driven evolution leads distinct species to spread over different niches. Consequently, the distribution of the various species at the end of the run unravels the location of the homogeneous regions on the original image. The method has been called selectionist relaxation because the segmentation emerges as a by-product of a relaxation process [27] mainly driven by selection.

As previously stated, the algorithm presented in [1] tunes a color images segmentation algorithm, namely Phoenix [28], by a chromosome formed by the program parameters, and not formed by image characteristics as in [19], [20], [21].

A complete survey on GA used in image processing is that one compiled by Alander [29].

III. GASe: Genetic Algorithm Segmentation environment

Using the same rationale as in [1] we adopted a GA for tuning the set of parameters of a range segmentation algorithm.

Different approaches to the tuning of parameters could be represented by Evolutionary Programming (EP) and Evolution Strategy (ES).

The first one places emphasis on the behavioral linkage between parents and their offsprings (the solutions). Each solution is replicated into a new population and is mutated according to a distribution of mutation types. Each offspring solution is assessed by computing its fitness. Similarly, the second one tries random changes in the parameters defining the solution, following the example of natural mutations.

Like both ES and EP, GA is a useful method of optimization when other techniques such as gradient descent or direct, analytical discovery are not possible. Combinatoric and real-valued function optimization in which the optimization surface or fitness landscape is "rugged", possessing many locally optimal solutions, are well suited for genetic algorithm.

We chose GA because it is a well-tested method in image segmentation, and a good starting point to explore the evolutionary framework.

Because of the universal model we have the possibility of changing the segmentation algorithm with few consequent changes in the GA code. These changes mainly involve the chromosome composition and the generation definition. The fitness evaluation has been modeled for the problem of range segmentation and can be kept constant as the reproduction model. This is one of the features of our proposal that we called GASe, or Genetic Algorithm Segmentation environment (introduced as GASP in [30]).

The main goal of GASe is to suggest a signature for a class of images, that is the best fitted set of parameters performing the optimal segmentation. In this way, when our system finds a good segmentation for an image or for a particular surface, we could that the same parameters will work correctly for the same class of images or for the same class of surfaces (i.e. all the surfaces presenting a big curvature ray).

A. The GASe architecture

In figure 1 we show the architecture of our system. Following the block diagram we see that an input image I_i is first segmented by a program s (range segmentation algorithm) with a parameter set Π_j^s producing a new image having labeled surface patches M_{ij}^s. All such segmented images are stored in a database that we call "Phenotype repository". Briefly, we may write

$$M_{ij}^s = segmentation(s, \Pi_j^s, I_i)$$

The quality of the segmentation process may be assessed by means of the so called fitness evaluation (in block "Genetic-based learning") computing a score F_{ij}^s by comparing the segmented image M_{ij}^s with the ground truth segmented image G_i.

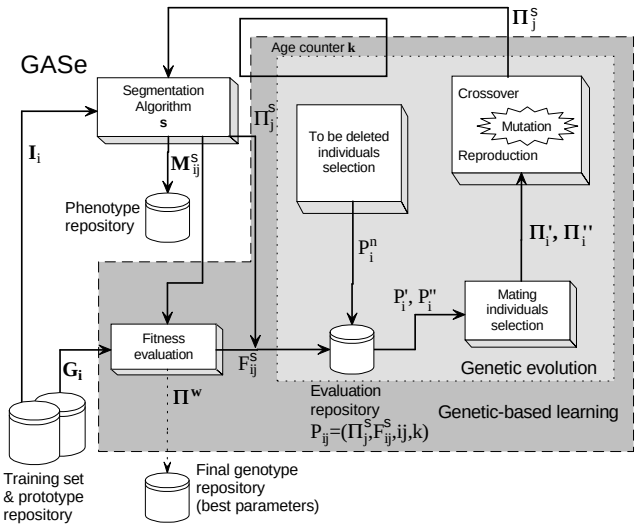


Fig. 1. GA architecture for range image segmentation

We assume our fitness function evaluates a cost therefor positively valued (or zero valued if the segmented image coincides exactly with the ground truth one). Thus

$$F_{ij}^s = fitness(M_{ij}^s, G_i), F_{ij}^s \geq 0$$

This process is fulfilled for all available images with different parameter sets. The sets that produce the best results (called Π^w) are stored in the so called "Final Genotype Repository" (if fitness function is under a given threshold). Once the score is assigned, a tuple P_{ij} containing the genotype, the score value, the phenotype identifier and the generation $(\Pi_j^s, F_{ij}^s, ij, k)$ is written in a database called "Evaluation Repository". The genetic computation selects two individuals to be coupled among the living ones ("Mating individuals selection"); these genotypes are processed by the "Crossover" block that outputs one or more offsprings that could be mutated. The generated individuals will be the new genotypes Π_j^s in the next generation step.

At the end of a generation a "To-be-deleted individuals selection" is performed. The decision on which individuals are to be erased from the evaluation repository is made by fixing a killing probability –p_k– depending on the fitness and the age of the individuals (their k value). If an individual has a score greater than p_k, the solution it represents will be no longer considered. In this way we have a limited number of evaluated points in the solution space.

B. GASe features

When building a GA some features have to be specifically designed. Among others we mention the *fitness function*, the *chromosome*, described in sections III-C and III-D, and the *crossover*.

The fitness function is a heuristic function that indicates to the GA whether an individual fits or not the environment. The chromosome is the data structure that contains the characters of the individuals. The crossover is the method that indicates how parents' characteristics are inherited by children. For this work

we used modified versions of Multiple Point Crossover [31] and Uniform Crossover [32], as described in [30].

C. Fitness function

The most critical step in the genetic evolution process is the definition of a reliable fitness function which ensures monotonousness with respect to the improvement provided by changing the segmentation parameters. The fitness function could be used both for comparing different algorithms and different parameter sets within the same algorithm. In [6] the problem of comparing range segmentation algorithms has been thoroughly analyzed, nevertheless the authors' evaluations take into account a number of separate performance figures and no global merit value is provided. More precisely the authors consider five figures that are functions of a precision percentage:

1) correct segmentation;
2) over-segmentation;
3) under-segmentation;
4) miss-segmentation;
5) noise-segmentation.

Conversely we are in the need of a single value which will then guide our feedback loop within the optimization process, and therefore we define a unique performance value specifically accounting for 1, 2 and 3; moreover errors due to missed regions or artifact regions are also considered, that group 4 and 5. In [33] and in [34] a function assigning a scalar to a segmentation is used. Particularly in [34] that function is the probability error between the ground truth and the machine segmented image. But such a way of assessing fitness is judged not suitable [6]. That should mean that a more robust way to have a scalar could be to order a vector of properties. Of course the ordering of vectors is not straightforward without using particular techniques; one of them could be to adopt a weighted sum of the components.

We define the fitness function as a weighted sum of a number of components:

$$F = w_1 C + w_2 H_u + w_3 H_o + w_4 U : \sum_{i=1}^{4} w_i = 1 \qquad (1)$$

where w_1, w_2, w_3 plus w_4 are tuned to weigh differently the single components.

The fitness takes into account two levels of errors (and therefore is a cost to be minimized); the former is a measure at pixel level computed with a pixel-by-pixel comparison; the latter is a measure at surface level considering the number of computed surfaces. At the pixel level, C is the Cost associated with erroneously segmented pixels and U accounts for Unsegmented pixels. At the surface levels we add two factors (Handicaps): one due to under-segmentation (H_u) and one due to over-segmentation (H_o).

Let G be the ground truth image, having N_G regions called R_{G_i} composed by P_{G_i} pixels, $i = 1..N_G$, and MS be the machine segmented image, having N_M regions called R_{M_j} composed by P_{M_j} pixels, $j = 1..N_M$. We define the *Overlap map* O, so that

$$O_{ij} = \#(\text{overlapping pixel ij of } R_{G_i} \text{ and } R_{M_j}) \qquad (2)$$

where $\#(.)$ indicates the number of $(.)$. The number of pixels with the same coordinates in the two regions is the value O_{ij}. The expression 2 could be written as $O_{ij} = R_{G_i} \cap R_{M_j}$. It is straightforward that if there is no overlap between the two regions $O_{ij} = 0$, while in case of complete overlap $O_{ij} = P_{G_i} = P_{M_j}$.

To compute the cost C, starting from O_{ij} we search the index $x_j \; \forall R_{M_j} : x_j = argmax_{i=1}^{N_G}(O_{ij})$.

$$C = \frac{\sum_{j=1}^{N_M}(P_{G_{x_j}} - O_{x_j j})}{N_M} \qquad (3)$$

In other words C should be a kind of distance between the real and the ideal segmentation at pixel level.

$$U = \sum_{i=1}^{N_M}(P_i - \sum_{j=1}^{N_G} O_{ij}) \qquad (4)$$

U accounts for the unlabeled pixels, i.e. those pixels that at the end of the process do not belong to any region (this holds only for the USF segmentation algorithm since the UB segmentation algorithm allocates all unlabeled pixels to the background region).

Then we can create another (boolean) *Matching map* with entries m_{ij} so that

$$m_{ij} = \begin{cases} 1 & \text{if } i = argmax_{j=1}^{N_M}(O_{ij}) \\ 0 & \text{otherwise} \end{cases} \qquad (5)$$

H_u is a "handicap" accounting for the number of under segmented regions (those which appear in the resulting image as a whole whilst were separated in the ground truth image);

$$H_u = k \cdot \#(R_{M_j} : \sum_{i=1}^{N_G} m_{ij} > 1, j = 1..N_M) \qquad (6)$$

In fact, in each row i of the Matching map only one entry is set to 1, while more entries in a column can be set to 1 if under-segmentation occurs and a segmented region covers more ground truth regions.

Finally, H_o is a "handicap" accounting for the number of over segmented regions (those which appear in ground truth image as a whole whilst were split in the resulting image);

$$H_o = k \cdot \#(R_{M_j} : \sum_{i=1}^{N_G} m_{ij} = 0, j = 1..N_M) \qquad (7)$$

H_o and H_u are both multiplied by a constant k just to enlarge the variability range.

Some results about the effectiveness of the adopted fitness function have been presented in [35].

D. Coding the chromosomes

One of the main tasks in GASe was to code the chromosome, i.e. to code the parameter set for a given segmentation algorithm.

To simplify the generation of new solution by a correct chromosome manipulation, we should use a binary coding but, since

TABLE I

USF PARAMETERS, MEANING AND VARIABILITY RANGE

Name	Name within code	Range	Meaning
N	WINSIZE	$2-12$	Window ray in which calculating normals
T_{point}	MAXPTDIST	$0-\infty$	Maximum point-to-point distance between pixel and 4-connected neighbor in region
T_{perp}	MAXPERPDIST	$0-\infty$	Maximum perpendicular distance between pixel and plane equation of region grown
T_{angle}	MAXANGLE	$0.0-180.0$	Maximum angle between normal of pixel and normal of region grown
T_{area}	MINREGPIX	$0-\infty$	Maximum number of pixels to accept or reject a region

TABLE II

UB PARAMETERS, MEANING AND VARIABILITY RANGE

Variable Name	Meaning	Variable type	Range*
Th_toleran	Curve segment accuracy	float	0.5–15.0
Th_length	Min curve segment length	int	3**
Th_jump	Min z distance for jump edges	float	1.0–20.0
Th_crease	Min angular distance for crease edges	float	0.0–180.0
Th_area	Min number of pixels for a valid surface	int	0–∞
Th_morph	Number of postprocessing morphological operators	float	1.0–3.0
Th_PRMSE	Plane region acceptance (RMSE)	float	0.1–10.0
Th_Pavgerr	Plane region acceptance (average error)	float	0.05–10.0
Th_CRMSE	Curve region acceptance (RMSE)	float	0.1–10.0
Th_Cavgerr	Curve region acceptance (average error)	float	0.05–10.0

*we decided to limit the range according to the observed lack of meaning of greater values when segmenting MSU/WSU images, so the shown limits are less than possible

**fixed by UB task force; we decided to allow a range 2 - 4

some genes (i.e. parameters) could assume real values, this coding did not suffice. So we decided to adopt an "extended" logical binary coding in order to represent real values with a fixed point code (with a defined number of decimal). Thus we define the symbol set as $\{0,1,$ *dot*$\}$ to allow a representation (of fixed but arbitrary precision) of the decimal of the number. The choice of a fixed precision could seem wrong, but we can consider that, beyond a certain precision, segmentation algorithm performances are not affected. We could have used a floating point representation of the chromosome, as suggested in [36], but in the case we studied, a fixed point representation seems to be sufficient. The binary strings are formed by the juxtaposition of BCD coded genes, memory consuming but giving accuracy to and from decimal conversion. The choice of extending the symbols set including *dot* was a help for visual inspection of the created population databases (listed in figure 1).

Our chromosome contains all the parameters (their meanings are listed in tables I and II) of the chosen segmentation algorithm. In this way the solution spaces considered are n-dimensional with $n = 5$ for USF and $n = 10$ for UB.

IV. EXPERIMENTAL RESULTS

Experiments carried out on GASe used as a benchmark the Michigan State University/Washington State University synthetic image database (that we will refer to as MSU/WSU database [37]) and a subset of the University of Bern real database (referred to as ABW). The tests performed are very time-consuming since each segmentation process is iterated for a single experiment many times (i.e. for each individual of the solution population and for each generation).

Since we tested our GA with both a fixed and random number of children crossover, according to [30] we have to use an alternative definition of *generation*. The term generation in GAs is often used as a synonym of the iteration step and is related to the process of creating a new solution. In our case a generation step is given by the results obtained in a fixed time slice. In this manner we can establish a time slice in function of the reference workstation; for instance with a standard PC (AMD Duron $700\mathrm{MHz}$) running Linux OS we could define the time slice as one minute of computation. In order to compare the efficacy and efficiency of results we will define a convergence trend maximum time to get the optimal solution in a given $MaxG$ generations.

A. Tuning the UB algorithm

The first experiment was the tuning of the UB segmentation algorithm [7]. This algorithm initially tries to detect the edges (jump and crease [38]) of the segmenting image by computing the "scan lines". After finding the candidates for area borders, it accomplishes an edge filling process. This segmentation algorithm is capable of segmenting curved-surfaces and the available version [39] can segment images of the GRF2-K2T database (named after the brand and model of the structured light scanner used). We used a version, slightly modified at the University of Modena, which is able to segment also synthetic images of the MSU/WSU database. A set of 35 images was chosen and a tuning task as in [6] was executed.

While the tuning done should provide very good results, it is our opinion that a training set should not be too large. We then chose a subset of 6 images as our training set. This set was input to GASe, and the resulting parameters set were used to segment the test set (formed by the remaining 29 images) and to find the most suitable set.

We fixed our generation in 1 minute and the maximum number of generations in 30. That is to say about 30 minutes of computation for every image of the training set. It took a total of about 3 hours to obtain 6 possible solutions, and to select the most suitable for the test set. During this time our algorithm performed about 10000 segmentations on the images. An exhaustive search should explore all the enormous space of solution (the space has 10 dimensions and one parameter potentially ranges from 0 to ∞) and all the instances of the test set. In our case, the exhaustive search was substituted by the GA-based search. Nevertheless, it is critical to test an individual on all images and measure the fitness as a function of the goodness over the whole training set.

As an acceptable approximation, to save computational time, we evaluated the fitness of every individual, applied on a single

TABLE III

PARAMETERS SETS FOR MODIFIED UB, AS TUNED BY ALGORITHM AUTHOR AND BY GASE

Parameter	Original opt. val.	GASe opt. val.
ThSegmToler	7.5	3.61
ThJump	10.0	4.55
ThCrease	30.0	36.78
ThPRMSE	1.11	0.51
ThPAvErr	1.07	0.21
ThCRMSE	1.11	0.57
ThCAvErr	1.09	0.45
ThPostprFact	2.0	1.79
ThSegmLen	3	2
ThRegArea	100	6

TABLE IV

AVERAGE FITNESS VALUES AS ALLOWED BY "ORIGINAL OPT. VAL." AND BY "GASE OPT. VAL."

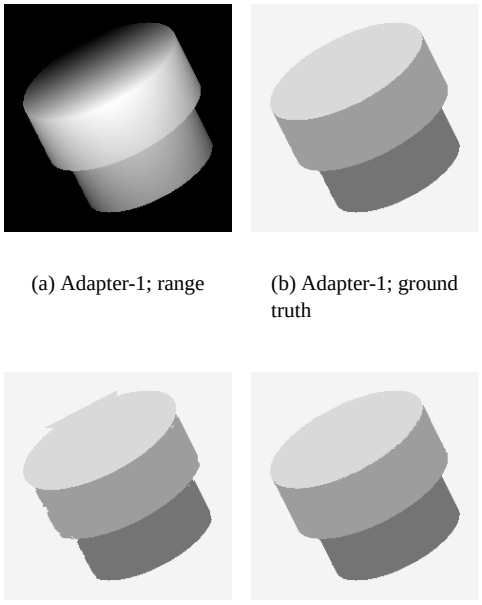
Parameters set	Average fitness
Original	15.96
GASe	15.04

image at a time. We assumed that, thanks to the genetic evolution, when the individual genotype becomes common in the population, it will be tested on different images. At the end the best scored individuals are tested on all images of the training set and the one that outperforms the others in average is selected as the best.

In table III we show the parameters used for this test. With "Original opt. val." we refer to the parameters tuned by the algorithm author, while with "GASe opt. val." we refer to those tuned by GASe. In table IV we show the average scores obtained in this test. Although the improvement could seem poor, it is not because of the presence of images with very different characteristics, which were not considered in the training set. As a matter of fact, the fitness improvement is in most of the cases of one or some units (see figures 2 and 3 where "original" and "GASe opt. val." are compared). The best improvement was of 11.26 points, while in one case only the GASe optimization generated a worst result with respect to the manual selection.

B. Tuning the USF algorithm

The second experiment was performed on the USF segmentation algorithm [6]. Based on a region growing strategy, it computes the normal vector for each pixel within a parametric-sized window. After that first computation it selects seed points on the basis of a reliability measure. From these seed points it accomplishes the region growing, aggregating surfaces until at least one of four parametric criteria is met. This segmentation algorithm has been tuned using a set of parameters proposed by its authors. As we can see in [6], the given results are very impressive, so we knew how difficult it will be to improve them. Nevertheless we performed the following experiment: given the original training set (10 images of the ABW database) we chose an image as our training set, and the other 9 as the test set. Then we compared the results on this subset to the corresponding former results on the same subset using the



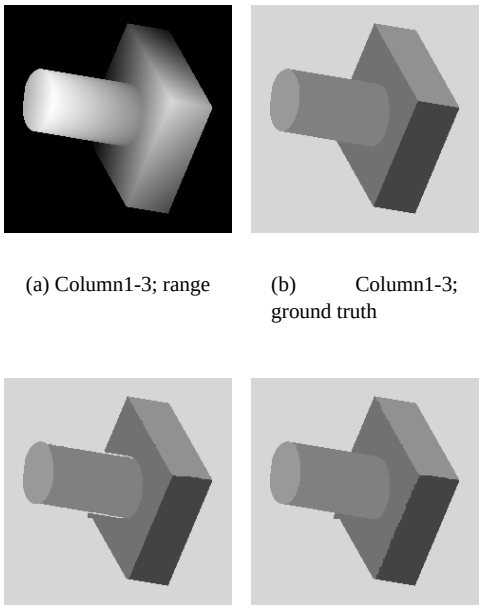
(a) Adapter-1; range

(b) Adapter-1; ground truth

(c) Adapter-1; Original opt. val. Fitness=4.61

(d) Adapter-1; GASe opt. val. Fitness=3.91

Fig. 2. Improvement of obtained segmentation for adapter-1



(a) Column1-3; range

(b) Column1-3; ground truth

(c) Column1-3; Original opt. val. Fitness=8.42

(d) Column1-3; GASe opt. val. Fitness=7.65

Fig. 3. Improvement of obtained segmentation for column1-3

TABLE V
PARAMETERS SETS FOR USF, AS TUNED BY ALGORITHM'S AUTHORS AND BY GASE

Parameter	Original opt. val.	GASe opt. val.
WINSIZE	10	9
MAXPTDIST	12.0	13.2
MAXPERPDIST	4.0	5.3
MAXANGLE	25.0	11.45
MINREGPIX	500	482

comparison tool presented in [6]. The comparison tool considers five types of region classification: correct detection, over-segmentation, under-segmentation, missed and noise. When all region classifications have been determined, a metric describing the accuracy of the recovered geometry is computed: any pair of regions R_1 and R_2 in the ground truth image, representing adjacent faces of the same object, have their angle A_n recorded in the truth data. If R_1 and R_2 are classified as correct detections, the angle A_m between the surface normals of their corresponding regions in the machine segmented image is computed. Then $\mid A_n - A_m \mid$ is computed for every correct detection classifications. The number of angle comparisons, the average error and the standard deviation are reported, giving and indirect estimation of the accuracy of the recovered geometry of the correctly segmented portion of image.

The set as tuned by GASe is in table V, and we refer to as "GASe opt. val.". In the same table are also included the parameters as tuned in [6], which are referred as "Original opt. val.". The results are not better than those presented in [6], but in a limited amount of time (we fixed the search in 15 generations) we reached a good result considering that the solution space was larger than that considered in [6]. Moreover no information is given about the time spent to select the solution space, while can be easily determined an average time to explore the whole solution space to select the "Original opt. val.".

In table VI we present the results determined by the two sets with a precision tolerance of 80% (see [6]). In figure 4 we show the plots corresponding to the experiment. The comparison tool provides five error measures, in addition to a measure of correctness. All these measures are related with a tolerance percentage. Plots of figures 4(a)–4(e) show the results on the training set of the "Original opt. val." (curve labeled as HE) vs. "GASe opt. val." (with label GA). The comparison is very interesting, especially considering that the heuristic selection was performed on a small solution space and tuned on all ten images, while the GASe one, although optimized by GAs, was tuned on a single image only.

In particular, figure 4(a) indicates that both parameter sets achieve the same number of correct instances over the training set, while figures 4(b) and 4(c) demonstrate that for problems of over and under-segmentation GASe and Original opt. val. have an opposite behavior since GASe produces less under-segmentation errors but higher over-segmentation. Finally the last two plots show that there is no noticeable difference in noise and miss segmentation.

V. DISCUSSION AND CONCLUSIONS

The segmentation of range images is a challenging problem both for the selection of the more appropriate algorithm (region growing, edge filling, clustering, etc.) and for the obtained accuracy. A variety of systems to perform this task have been presented in the literature (we remember [6], [15]), and all of them need an accurate parameters tuning, according to the image characteristics.

A tool to compare results was proposed in [6], and it has been used to address the parameters tuning (as in [2], [3], [4]) using only one of the given measures. The tuning methods are based either on careful selection or on solution space-partitioning search, which limits the dimensions of the solution space.

We proposed an automated search method, based on genetic algorithms, that allows us to search a large solution space, while requiring a manageable amount of computation time (according to the chosen segmentation algorithm). To address the search we used a fitness function that combines different measures given by the comparison tool (although using a different source code). We thus implemented a system, called GASe (Genetic Algorithm Segmentation environment), to test different segmentation algorithms, namely UB and USF.

We saw that for the UB we obtained excellent results, improving segmentation quality and the speed of segmentation. For the USF we obtained reasonable results, similar to the one proposed by the authors, but without having any knowledge about the nature of the parameters. In fact, GAs start from random values of the parameter set and are able to reach a similar solution in relatively few generations. Finally, embedded in GASe and as a stand alone tool, we proposed an algorithm to robustly award a scalar value to a segmentation.

We believe this work provides the basis to design a wizard (or expert system) helping human operators in segmenting images. Our final aim is to build an interactive system, that, after an unsupervised training time, will help human operators in the task of obtaining good segmentations. The expert system will provide the framework for the operator to decide the parameters to segment a single or a subset of surfaces in a complex scene (as done in [40]).

REFERENCES

[1] B. Bhanu, S. Lee, and J. Ming, "Adaptive Image Segmentation Using a Genetic Algorithm," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 25, no. 12, pp. 1543–1567, 1995.

[2] J. Min, M. W. Powell, and K. W. Bowyer, "Progress in Automated Evaluation of Curved Surface Range Image Segmentation," in *Proc. of Int. Conf. on Pattern Recognition*, pp. 644–647, Barcelona, 2000.

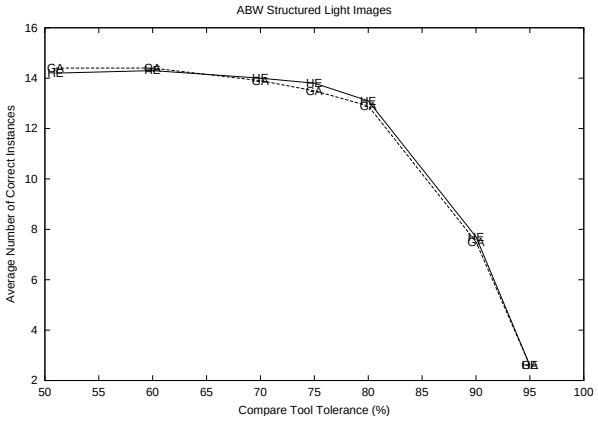
[3] J. Min, M. W. Powell, and K. W. Bowyer, "Automated Performance Evaluation of Range Image Segmentation," in *IEEE Workshop on Applications of Computer Vision*, pp. 163–168, Palm Spring, 2000.

[4] J. Min, M. W. Powell, and K. W. Bowyer, "Objective, Automated Performance Benchmarking of Region Segmentation Algorithms," in *Int. Conf. on Computer Vision*, Barcelona, 2000.

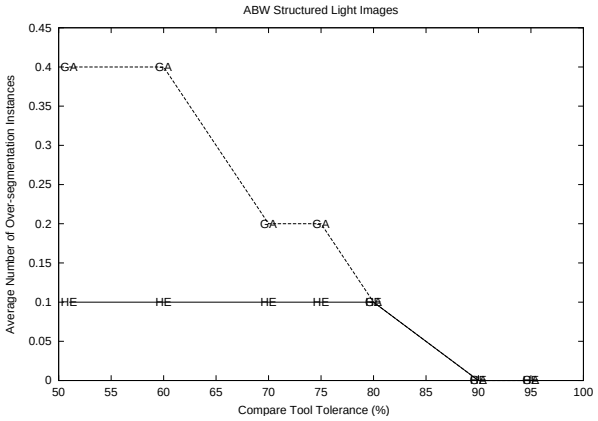
[5] X. Jiang, "An Adaptive Contour Closure Algorithm and Its Experimental Evaluation," *IEEE Trans. on PAMI*, vol. 22, pp. 1252–1265, November 2000.

[6] A. Hoover, G. Jean-Baptiste, X. Jiang, *et al.*, "An Experimental Comparison of Range Image Segmentation Algorithms," *IEEE Trans. on PAMI*, vol. 18, pp. 673–689, July 1996.

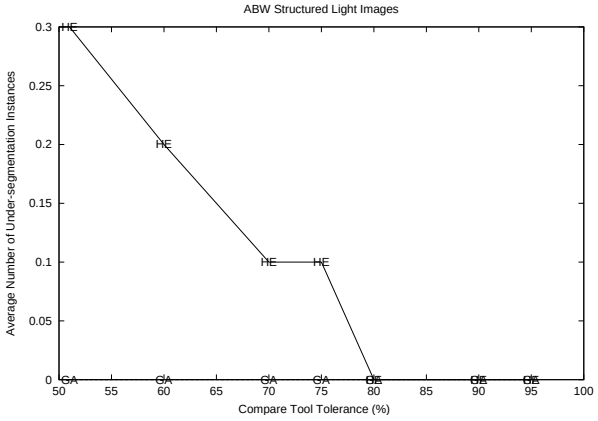
[7] X. Jiang and H. Bunke, "Edge Detection in Range Images Based on Scan Line Approximation," *Computer Vision and Image Understanding*, vol. 73, pp. 183–199, February 1999.



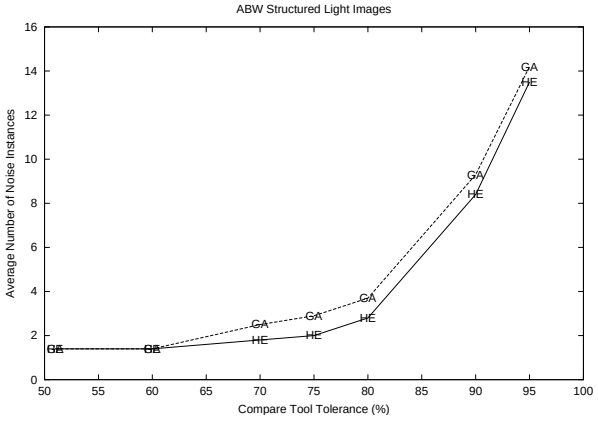
(a) Average correct detections on 10 ABW images



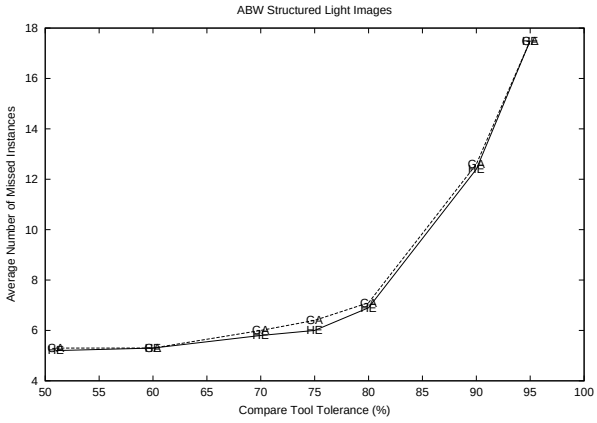
(b) Average over-segmentations on 10 ABW images



(c) Average under-segmentations on 10 ABW images



(d) Average noise regions on 10 ABW images



(e) Average missed regions on 10 ABW images

Fig. 4. Results, as measured by the comparison tool, obtained by the “Original opt. val.” (labeled “HE”) and “GASe opt. val.” (labeled “GA”) on ten images of the ABW database

TABLE VI

AVERAGE RESULTS OF USF SEGMENTATION ALGORITHM WITH ORIGINAL AND GASE OPT. VAL. ON 10 ABW IMAGES AT 80% OF COMPARE TOLERANCE (WE REMEMBER THAT TOOL MEASURES SEGMENTATION ALGORITHM PERFORMANCES WITH RESPECT TO A CERTAIN PRECISION TOLERANCE, RANGING FROM 51 TO 95%)

Parameters set	GT regions	Correct detection	Angle diff. (std. dev.)	Over-segmentation	Under-segmentation	Missed	Noise
Original	20.1	13.1	1.24° (0.96)	0.1	0.0	6.9	2.8
GASe	20.1	12.9	1.27° (0.99)	0.1	0.0	7.1	3.7

[8] X. Jiang, H. Bunke, and U. Meier, "High-level feature based range image segmentation," *Image and Vision Computing*, pp. 817–822, July 2000.

[9] Y. Zhang, H. Sari-Sarraf, J. Price, and M. A. Abidi, "Impact of Intensity Edge Map on Segmentation of Noisy Range Images," in *SPIE #3958: Three Dimensional Image Capture and Applications III*, pp. 260–269, 1990.

[10] I. S. Chang and R.-H. Park, "Segmentation based on fusion of range and intensity images using robust trimmed methods," *Pattern Recognition*, vol. 34, pp. 1951–1962, October 2001.

[11] M. Baccar, L. A. Gee, and M. A. Abidi, "Reliable Location and Regression Estimates with Application to Range Image Segmentation," *J. of Mathematical Imaging and Vision*, vol. 11, pp. 195–205, December 1999.

[12] H. D. Cheng, X. H. Jiang, Y. Sun, and J. Wang, "Color image segmentation: advances and prospects," *Pattern Recognition*, vol. 34, pp. 2259–2281, December 2001.

[13] X. Jiang, K. Bowyer, Y. Morioka, *et al.*, "Some further results of experimental comparison of range image segmentation algorithms," in *Proc. of Int. Conf. on Pattern Recognition*, vol. 4, pp. 877 – 881, 2000.

[14] M. W. Powell, "Comparing curved surface range image segmenters," in *Proc. of Int. Conf. on Computer Vision*, pp. 286–291, 1998.

[15] H. Frigui and R. Krishnapuram, "A Robust Competitive Clustering Algorithm With Applications in Computer Vision," *IEEE Trans. on PAMI*, vol. 21, pp. 450–465, May 1999.

[16] D. E. Goldberg, *Genetic algorithms in search optimization and machine learning*. Reading, MA: Addison-Wesley, 1989.

[17] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolutionary Computation*. New York, N.Y.: Springer Verlag, 2^{nd} ed., 1994.

[18] M. Srinivas and L. M. Patnaik, "Genetic Algorithms: a Survey," *IEEE Computer*, vol. 27, no. 6, pp. 17–27, 1994.

[19] X. Yu, T. D. Bui, and A. Krzyżak, "Robust Estimation for Range Image Segmentation and Reconstruction," *IEEE Trans. on PAMI*, vol. 16, pp. 530–538, May 1994.

[20] D. N. Chun and H. S. Yang, "Robust Image Segmentation Using Genetic Algorithm with a Fuzzy Measure," *Pattern Recognition*, vol. 29, no. 7, pp. 1195–1211, 1996.

[21] P. Andrey and P. Tarroux, "Unsupervised Image Segmentation Using a Distributed Genetic Algorithm," *Pattern Recognition*, vol. 27, no. 5, pp. 659–673, 1994.

[22] M. Yoshimura and S. Oe, "Evolutionary segmentation of texture image using genetic algorithms towards automatic decision of optimum number of segmentation areas," *Pattern Recognition*, vol. 32, pp. 2041–2054, December 1999.

[23] D.-C. Tseng and C.-C. Lai, "A genetic algorithm for MFR-based segmentation of multi-spectral textured images," *Pattern Recognition Letters*, vol. 20, pp. 1499–1510, December 1999.

[24] J. Besag, "On the Statistical Analysis of Dirty Pictures," *J. of the Royal Stat. Soc. B*, vol. 48, pp. 259–302, December 1986.

[25] S. Cagnoni, A. B. Dobrzeniecki, R. Poli, and J. C. Yanch, "Genetic algorithm-based interactive segmentation of 3D medical images," *Image and Vision Computing*, vol. 17, no. 12, pp. 881–895, 1999.

[26] P. Andrey, "Selectionist relaxation: genetic algorithms applied to image segmentation," *Image and Vision Computing*, vol. 17, pp. 175–187, March 1999.

[27] L. S. Davis and A. Rosenfeld, "Cooperating Processes for Low-Level Vision: A Survey," *Artificial Intelligence*, vol. 17, no. 1-3, pp. 245–263, 1981.

[28] K. I. Laws, "The Phoenix Image Segmentation System: Description and Evaluation." SRI Int. Technical Note No. 289, December 1982.

[29] J. T. Alander, "Indexed bibliography of genetic algorithms in optics and image processing," Tech. Rep. 94-1-OPTICS, University of Vaasa, Department of Information Technology and Production Economics, 2000.

[30] L. Cinque, R. Cucchiara, S. Levialdi, S. Martinz, and G. Pignalberi, "Optimal Range Segmentation Parameters Through Genetic Algorithms," in *Proc. of Int. Conf. of Pattern Recognition*, pp. 474–477, Barcelona, 2000.

[31] L. J. Eshelman, A. Caruana, and J. D. Schaffer, "Biases in the Crossover Landscape," in *Proc. of Int. Conf. on Genetic Algorithms* (J. D. Schaffer, ed.), pp. 86–91, 1989.

[32] G. Syswerda, "Uniform crossover in genetic algorithms," in *Proc. of the Int. Conf. on Genetic Algorithms* (J. Schaffer, ed.), pp. 2–9, 1989.

[33] M. D. Levine and A. M. Nazif, "An Experimental Rule Based System for Testing Low Level Segmentation Strategies," in *Multicomputers and Image Processing: Algorithms and Programs* (K. Preston and L. Uhr, eds.), pp. 149–160, Academic Press, 1982.

[34] Y. W. Lim and S. U. Lee, "On the Color Image Segmentation Algorithm Based on the Thresholding and the Fuzzy C-Means Techniques," *Pattern Recognition*, vol. 23, pp. 935–952, September 1990.

[35] L. Cinque, R. Cucchiara, S. Levialdi, and G. Pignalberi, "A methodology to award a score to range image segmentation," in *Proc. of Int. Conf. on Pattern Recognition and Information Processing*, pp. 171–175, Minsk, 2001.

[36] F. Herrera, M. Lozano, and J. L. Verdegay, "Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis," *Artificial Intelligence Review*, vol. 12, no. 4, pp. 265–319, 1998.

[37] "MSU/WSU range image database." http://sampl.eng.ohio-state.edu/~sampl/data/3DDB/RID/index.htm, 2002.

[38] R. Hoffman and A. K. Jain, "Segmentation and Classification of Range Images," *IEEE Trans. on PAMI*, vol. 9, pp. 608–620, September 1987.

[39] "Range image segmentation comparison project." http://marathon.csee.usf.edu/range/seg-comp/results.html, 2002.

[40] L. Cinque, R. Cucchiara, S. Levialdi, and G. Pignalberi, "A Decision Support System for Range Image Segmentation," in *Proc. of Int. Conf. Digital Image Processing and Control in Extreme Situations*, pp. 45–50, Minsk, 2002.