N. Pavešić, H. Niemann, S. Kovačič, F. Mihelič (Eds.)

Speech and Image Understanding

Proceedings of 3rd Slovenian-German and 2nd SDRV Workshop
April 24-26, 1996, Ljubljana, Slovenia

IEEE Slovenia Section

Speech and Image Understanding, Proceedings of 3rd Slovenian-German and 2nd SDRV Workshop, April 24-26, 1996, Ljubljana, Slovenia, pp. 331-340.

Segmentor: An Object-Oriented Framework for Image Segmentation *

Aleš Jaklič, Aleš Leonardis and Franc Solina

Computer Vision Laboratory
Faculty of Computer and Information Science, University of Ljubljana
Tržaška cesta 25, 1001 Ljubljana, Slovenia
Tel. +386 (61) 1768 381, Fax +386 (61) 1264 630
alesj@razor.fri.uni-lj.si

Abstract

Segmentor is an object-oriented framework for image segmentation based on a recover-and-select paradigm. It uses data abstraction, inheritance and polymorphism to simplify and speed up application of the paradigm to different image domains using various image models. The paper presents the object-oriented analysis of the problem domain, followed by the design and implementation of the framework. The case of range image segmentation with superquadrics is presented to illustrate the use of the framework.

1 Introduction

Application of recover-and-select paradigm [6] to image segmentation problem in computer vision produced promising results in segmentation of

- range images: in terms of biquadratic surface models [6] and superquadric volumetric models [9, 8],
- edge images: in terms of subset of conics curve models [7].

In order to demonstrate the general value of the paradigm we have to experimentally verify its applicability to various image domains (range images, gray images, edge images, etc.) and task domains (object recognition, object manipulation, navigation, geometric

*This work was supported by the Ministry of Science and Technology of Republic of Slovenia (Project J2-6187), European Union Copernicus Program (Grant 1068 RECCAD), and by U.S. - Slovene Joint Board (Project #95-158).

modeling, etc.). This requires experimentation in finding the appropriate key elements of the paradigm, namely: parametric image models, distance functions between models and image values, model parameter estimation algorithms, algorithms for description growing, functions measuring the quality of description, i.e. objective function, and description selection algorithms.

Since the recover-and-select paradigm is an iterative procedure, visualization of iterative steps during segmentation process, in addition to the quantitative error measures, can provide the experimenter the critical feedback needed to asses the quality and suitability of chosen elements of the paradigm. Visualization is also of utmost importance in testing and debugging experimental elements of the paradigm as well as the framework itself.

All current implementations of the paradigm use C programming language within UNIX operating system and X Window System. They do not allow for easy experimentation. This is primarily due to a purely functional software design, encouraged by C semantics, and low level of abstractions.

The primary motivation of this work is to design and implement an object oriented framework to *support* experimentation with the paradigm. By casting the recover-and select paradigm into an object oriented framework, we also expect to make the paradigm more concise and general. That might also lead to application of the paradigm to other disciplines beside computer vision.

We selected C++ as an object oriented programming language [14] for the implementation and X Window System to implement the user interface. The former decision allows us to encapsulate and reuse existing software for numerical optimization written in C.

2 Object-Oriented Analysis of Recover-and-Select Paradigm

In this section we describe the main classes of the problem domain, at the highest level of abstraction, together with the examples of concrete classes for particular image domains. The examples naturally suggest the class hierarchy. Also the dynamic behavior of description objects is presented. It is expressed in terms of abstract classes, suggesting the use of polymorphism for concrete class objects. The class diagrams are presented in Booch notation [1].

World is a finite set of points from a finite dimensional vector space, with defined distance function $d(x, y)$ between the points. The distance function of a point from the subset V of the world is defined as

$$d(x, V) = \min_{y \in V} d(x, y). \tag{1}$$

The ϵ neighborhood of subset V is defined as a set of points, such that the distance of a point from the subset is less or equal ϵ, and the points are not members of the subset V

$$N_\epsilon(V) = \{x : d(x, V) \leq \epsilon \wedge x \notin V\}. \tag{2}$$

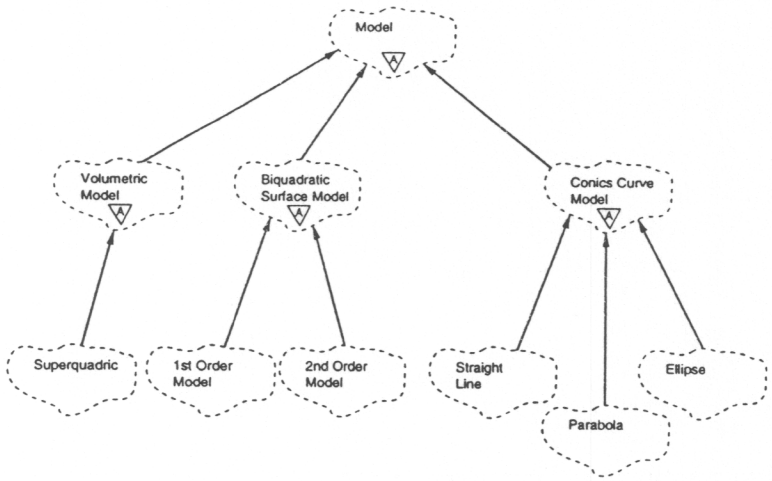

Figure 1: Class diagram of **models**

Model is an abstract entity with a defined distance $d(M, x)$ between the model and the point from the world. The distance between the model and the subset V of the world is in turn defined as

$$d(M, V) = \sum_{x \in V} d(M, x). \tag{3}$$

Model is usually parameterized with a finite number of parameters. We denote the parametric model as $M(\mathbf{p})$, where $\mathbf{p}$ is a finite dimensional vector of parameters with dimension $\dim(\mathbf{p})$. To use the model in the recover-and-select paradigm one must be able to find the parameters $\mathbf{p}_{min}$ that will minimize the distance between the model and the given subset V of the world

$$d(M(\mathbf{p}_{min}), V) = \min_{\mathbf{p}} d(M(\mathbf{p}), V). \tag{4}$$

Models do not exist in isolation from the world. As their name implies, they model the subsets of the world.

Description is a pair (V, M) of subset from the world and the model that describe this subset. The quality of subset representation is measured by description error ξ, which equals the distance between the model and the subset

$$\xi = d(M, V). \tag{5}$$

Descriptions are objects that iteratively change until they reach their final state. The iteration consists of:

1. forming the neighborhood of the description,

$$N(M, V) = \{x : x \in N_\epsilon \wedge d(M, x) \leq \gamma\} \tag{6}$$

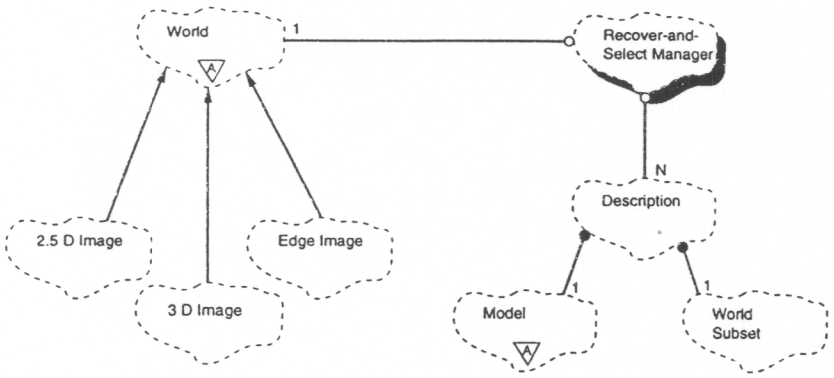

Figure 2: Class diagram of **world, description, and recover-and-select class utility**

if the set is empty the description enters a terminal state, from where it cannot evolve any more

2. finding the model for the union of the neighborhood and the world subset of the description, that minimizes the distance between the union and the model

$$d(M(\mathbf{p}_{min}), V \cup N(M,V)) = \min_{\mathbf{p}} d(M(\mathbf{p}), V \cup N(M,V)). \tag{7}$$

3. if the distance of the model from the union is lower than a threshold *max model error*, the description is updated by the union and the new model.

4. otherwise the description might try to use another type of model at step 2, if all model types have been already tried, the description enters the terminal state and cannot evolve any more

Recover-and-select paradigm initiates a set of descriptions in the world, and then allow them to iteratively evolve. Since the evolution of descriptions is computationally expensive the descriptions that do not model the world well enough are discarded during a selection step. The selection of individual descriptions is based on the solution of the Quadratic Boolean Problem, where function $F(\mathbf{d})$ that measures the quality of the overall description of the world is maximized

$$F(\mathbf{d}) = \mathbf{d}^T \mathbf{Q} \mathbf{d} = \mathbf{d}^T \begin{bmatrix} c_{11} & \dots & c_{1N} \\ \vdots & & \vdots \\ c_{N1} & \dots & c_{NN} \end{bmatrix} \mathbf{d} \ . \tag{8}$$

Where $\mathbf{d}^T = [d_1, d_2, \dots, d_N]$ is a vector of presence variables, d_i having the value 1 for the presence and 0 for the absence of the description D_i in the overall description. The

diagonal terms of the matrix $\mathbf{Q}$ express the cost-benefit value of a particular description D_i,

$$c_{ii} = \mathrm{K}_1|V_i| - \mathrm{K}_2\xi_i - \mathrm{K}_3 \dim(\mathbf{p}_i) \ , \tag{9}$$

where K_1, K_2, K_3 are weights which can be determined on a purely information-theoretical basis [6]. The off-diagonal terms handle the interaction between the overlapping descriptions

$$c_{ij} = \frac{-\mathrm{K}_1|V_i \cap V_j| + \mathrm{K}_2\xi_{ij}}{2} \ . \tag{10}$$

where the mutual description error equals

$$\xi_{ij} = \max(d(M_i, V_i \cap V_j), d(M_j, V_i \cap V_j)). \tag{11}$$

Maximizing the objective function $F(\mathbf{m})$ belongs to the class of problems known as combinatorial optimization problems. Since the number of possible solutions increases exponentially with the size of the problem, it is usually not tractable to explore them exhaustively. A few approaches to the problem are described in [6] and a fast greedy algorithm with worst case time complexity of $O(n^2)$ is presented in [3]. For this analysis only the formation of the matrix $\mathbf{Q}$ is important since it involves the interaction of descriptions to calculate the off-diagonal terms.

3 User Interface

We will describe a user interface through a usage scenario used for adaptation of the paradigm to superquadric models. First the superquadric class was derived from the abstract model class and the pure virtual functions were implemented. The growing algorithm was then tested by manually placing seeds on different parts of the image and and by observing the growth of those seeds. The maximum point distance and maximum description error were experimentally tested. We observed that in some cases the models stopped growing due to our simple growing strategy, that always estimated the initial superquadric parameters for the non-linear iterative minimization from the data, not using the recovered model parameters from the previous step. This was then alleviated by recovering two models, one recovered from initial parameters derived directly from the data and another one recovered from the parameters of the model from previous iteration. Once we had the basic growing algorithm working, we proceeded to automatic grid like placement of seeds. Then appropriate constants for the selection procedure were determined. An option to save and restore models from files was instrumental in this experimentation to decouple the selection phase from the growing phase. This part of experimentation involves frequent user interaction with the models to examine their parameters.

After having a few test images successfully segmented, we proceeded to a larger experiments done in a batch mode. To support such experimentation the crucial decision was to provide a set of orthogonal commands and a simple terminal input. The segmentation is performed as a sequence of basic commands, which together with terminal input and

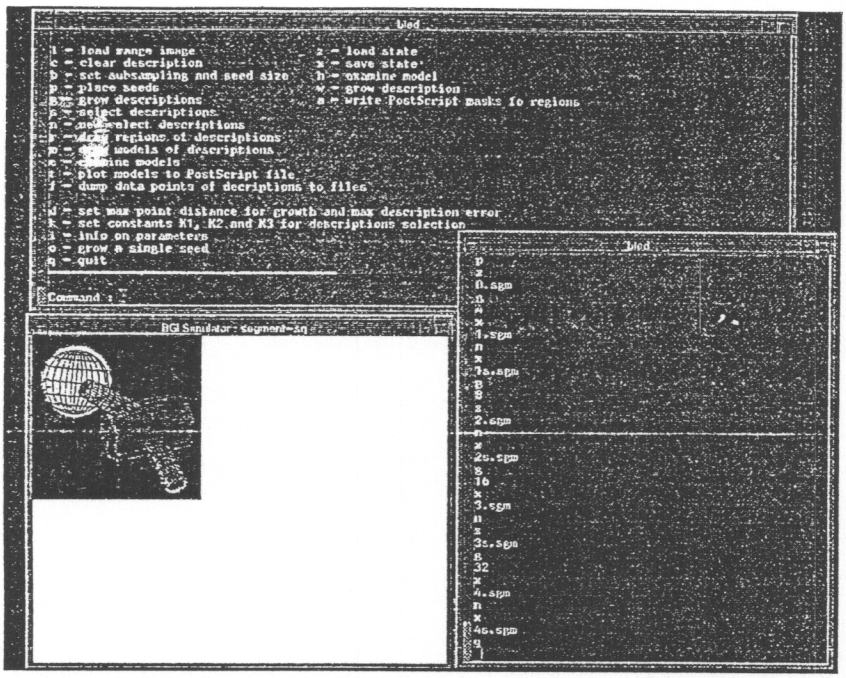

Figure 3: A screen snapshot of a Segmentor session

standard input redirection provides us with a tool for large batch experiments, where intermediate results are stored to files for later use.

4 Experimental Results

This section presents experimental parameters and results for the case of superquadric models. The Figure 4 presents a sequence of the original range image overlaid with the superqudrics during the iteration. The final part of the sequence presents subsets of range data points corresponding to individual models in the final description. The error distributions for those models are presented in Figure 5.

To reduce the computational burden mostly dictated by the numerical minimization during recovery of superquadrics, the range image of size 256 x 256 pixels was subsampled at factor 4, initial size of the seed was 16 x 16 pixels (effectively 4 x 4 after subsampling). The maximum point distance was set to 6.0, because of the subsampling and maximum allowed model error was set to 2.0. To prevent numerical degeneracies in minimization of the superquadric fitting function for subsets of points from planar regions, parameters determining the size of superquadrics were limited from bellow at 1.0 by a projection method (for details see [8]). The constants used during the selection were $K_1 = 1.0$,

$K_2 = 0.3$, $K_3 = 0.5$. The growth was iterated 4 times followed by a selection. The growth for the remaining models was iterated 8 times followed by a selection. This procedure of doubling the number of growth iterations followed by a selection was performed until there were any descriptions that could grow.

The processing times for the range image in Figure 4 on different workstations are listed in Table 1. However, processing time is not critical since individual models could be recovered in parallel. Besides, the computation of the superquadric fitting function and its derivatives is independent for each range point and can also be parallelized in a straightforward way.

Pentium 133 MHz, Linux 2.0.0	12 min.
HP 715/100, HP-UX 9.05	13 min.
HP 715/50, HP-UX 9.05	22 min.
Intel 486 DX2 66 MHz, Linux 1.2.13	59 min.

Table 1: The total processing times to produce results presented in Figure 4

5 Conclusions and Further Work

The framework was successfully used in applying the recover-and-select paradigm to segmentation of range images using superquadrics as models. Object-oriented analysis of the problem domain of image segmentation using recover-and-select paradigm not only produced the base for the design and implementation of the framework but also produced a more concise and generalized description of the paradigm itself.

Though object-oriented methods and programming are gradually appearing in the experimental computer vision software [2, 4, 5, 11, 12], they are rather an exception to the rule of procedural programming. Existing software environments implemented in non object-oriented programming languages like C [13], focus mainly on data representation and less on the algorithms that process the data. The later being completely separated from the data. Although they do provide a modest level of data abstraction and might even show some object-oriented design, they fail to use inheritance and polymorphism as efficient means for integration of new functionality into existing software environments.

We advocate that experimental computer vision software should be build in a form of object-oriented frameworks. Designing and implementing a framework is an iterative process and takes more time to build a framework than a procedural library. However, the long term benefits in form of software reuse outweigh apparent short term costs. For guidelines how to build frameworks see [1, 10].

We used the framework to build two independent programs for segmentation of range images in terms of planar surface patches and in terms of superquadrics as volumetric models. The work on combining both kinds of models simultaneously is in progress and presents the framework with no challenges. A large part of the framework is also being

used in a development system for integration of range images acquired from different viewpoints.

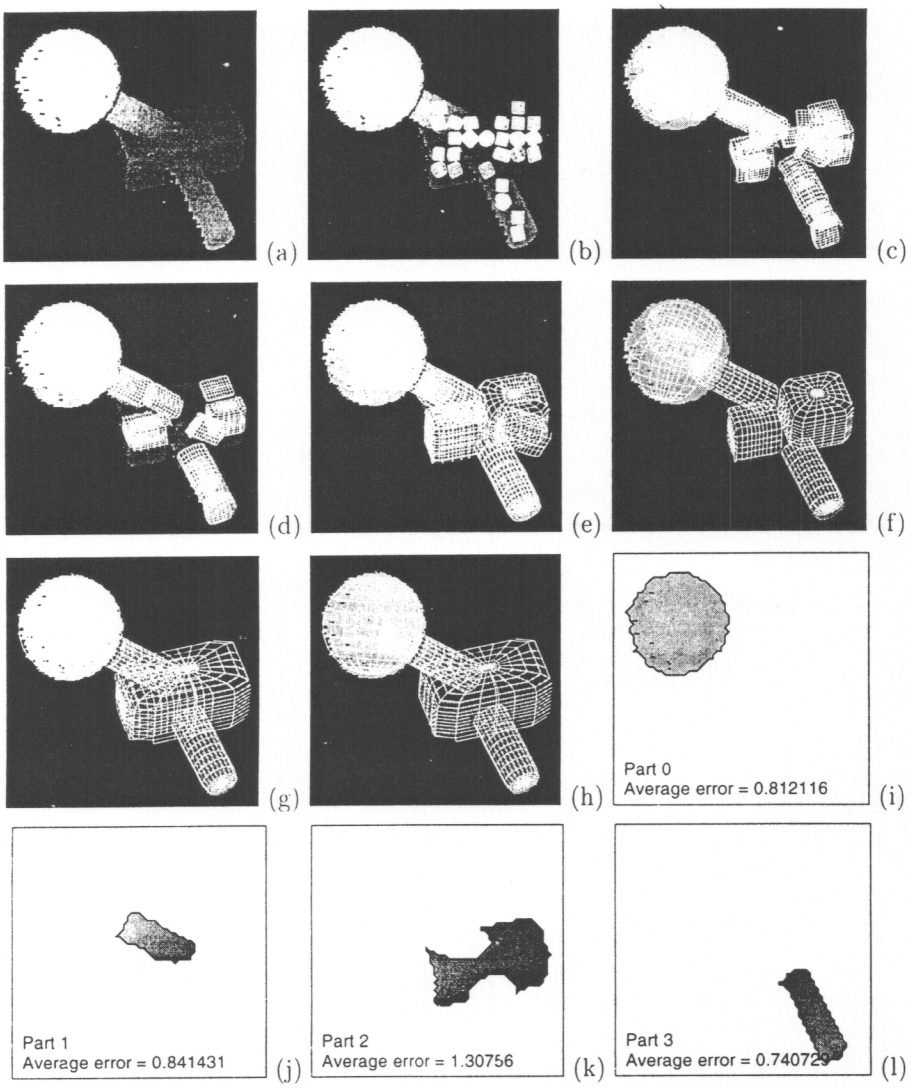

Figure 4: Sequence of images representing the steps in segmentation of a range image in terms of superellipsoids a) - h), subsets of the range image corresponding to individual descriptions i) - l).

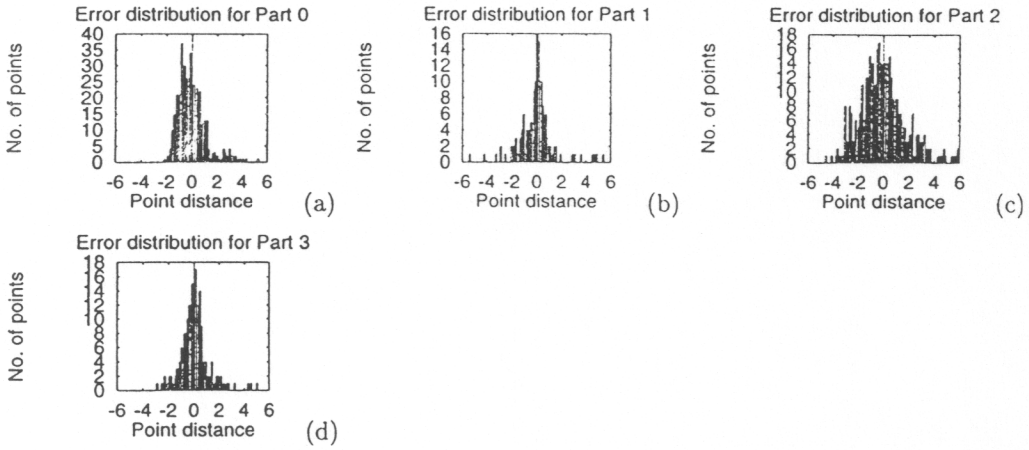

Figure 5: Error distributions for the descriptions from Figure 4. Note that a point inside a superquadric is assigned a negative distance, with the absolute value equal to the distance. A point outside the superquadric is assigned the real distance value.

References

[1] G. Booch. *Object-oriented Analysis and Design.* Benjamin/Cummings, second edition, 1993.

[2] M. Flickner, M. Lavin, and S. Das. An object-oriented language for image and vision execution (olive). In *Proceedings 10th IAPR International Conference on Pattern Recognition*, pages 561–571, 1990.

[3] A. Jaklič, A. Leonardis, and F. Solina. Segmentor: An object-oriented framework for image segmentation. Technical Report LRV-96-2, Computer Vision Laboratory, University of Ljubljana, Faculty of Computer and Information Science, 1996.

[4] D. Koelma. A software Environment for Image Interpretation. Ph.D. Thesis, University of Amsterdam, 1996

[5] D. T. Lawton and D. M. Mead. A modular object oriented image understanding environment. In *Proceedings 10th IAPR International Conference on Pattern Recognition*, pages 611–616, 1990.

[6] A. Leonardis. *Image Analysis Using Parametric Models.* PhD thesis, Faculty of Electrical Engineering and Computer Science, University of Ljubljana, 1993.

[7] A. Leonardis and R. Bajcsy. Finding parametric curves in an image. In G. Sandini, editor, *Proceedings of The Second European Conference on Computer Vision*, pages 653–657. Springer-Verlag, 1992.

[8] A. Leonardis, A. Jaklič, and F. Solina. Superequadrics for segmenting modeling and range data Technical Report LRV-96-1, Computer Vision Laboratory, University of Ljubljana, Faculty of Computer and Information Science, 1996.

[9] A. Leonardis, F. Solina, and A. Macerl. A direct recovery of superquadric models in range images using recover-and-select paradigm. In *Proceedings of Third European Conference on Computer Vision, Vol. I*, pages 309–318, 1994.

[10] A Taglient White Paper. Building object-oriented frameworks. Technical Report http://www.taligent.com/building-oofw.html, Taligent, Inc., 1994.

[11] D. Paulus, H. Niemann. *Object–Oriented Programming for Image Analysis*, in J. Menon (Hrsg.): *Current Topics of Pattern Recognition Research*, Vol. 1 *Research Trends*, India, 1996, pp. 185–204.

[12] J. Plomp. An object oriented representational system for image features and their relations. In *Proceedings 11th IAPR International Conference on Pattern Recognition*, pages 518–520, 1992.

[13] A. R. Pope and D. G. Lowe. Vista: A software environment for computer vision research. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 768–772, 1994.

[14] B. Stroustrup. *The C++ Programming Language.* Addison-Wesley, second edition, 1993.