JESS: Java Extensible Snakes System

Tim McInerney[a], M. Reza Akhavan Sharif[b] and Nasrin Pashotanizadeh[b]

[a]Dept. of Computer Science, [b]Dept. of Electrical Engineering;
Ryerson University, 350 Victoria St., Toronto, ON M5B 2K3, Canada

ABSTRACT

Snakes (Active Contour Models) are powerful model-based image segmentation tools. Although researchers have proven them especially useful in medical image analysis over the past decade, Snakes have remained primarily in the academic world and they have not become widely used in clinical practice or widely available in commercial packages. A number of confusing and specialized variants exist and there has been no standard open-source implementation available. To address this problem, we present a Java Extensible Snakes System (JESS) that is general, portable, and extensible. The system uses Java *Swing* classes to allow for the rapid development of custom graphical user interfaces (GUI's). It also incorporates the *Java Advanced Imaging (JAI)* class library, which provide custom image preprocessing, image display and general image I/O. The Snakes algorithm itself is written in a hierarchical fashion, consisting of a general Snake class and several subclasses that span the main variants of Snakes including a new, powerful, robust subdivision-curve Snake. These subclasses can be easily and quickly extended and customized for any specific segmentation and analysis task. We demonstrate the utility of these classes for segmenting various anatomical structures from 2D medical images. We also demonstrate the effectiveness of JESS by using it to rapidly build a prototype semi-automatic sperm analysis system. The JESS software will be made publicly available in early 2005.

Keywords: Segmentation, Snakes, Java, image processing system

1. INTRODUCTION

Snakes[1] (Active Contour Models) are powerful model-based image segmentation tools for the efficient and accurate segmentation of anatomical structures from medical images. To our knowledge, no general, open-source, portable Snakes software has emerged as a standard and we believe this has been a significant factor in the lack of wide-spread use of the powerful Snakes algorithm in clinical radiological practice. Although the original Snakes algorithm was developed over a decade ago, a plethora of specialized variants were developed resulting in well over one hundred academic papers. The software for these specialized versions was typically not made available and there was no standard method for creating portable graphical user interfaces or handling the myriad image modalities and image formats. To address this problem, we have created a Java-based open source Snakes package, known as JESS (Java Extensible Snakes System), for the rapid creation of customized model-based image segmentation and analysis systems.

In this paper, we will describe the implementation and structure of JESS. We will present several segmentation examples using the various Snakes classes supported in JESS. We will then describe the construction of a prototype semi-automatic sperm analysis system using JESS. This paper will focus on the system implementation and features of JESS. For mathematical details of Snakes, we refer the reader to several references.[1–3]

2. BACKGROUND

Active contour models, or *Snakes*, were originally introduced by Kass, Witkin, and Terzopoulos[1] in 1987. In the general case, Snakes are energy minimizing splines that are guided by user constraint forces (via a mouse or some other input device) and attracted to features such as lines and edges by external image forces. Internal forces are used to constrain the shape and smoothness of Snakes. The classical Snakes model is typically initialized by tracing a rough curve near the target object boundary. A Snake is created from this curve which then locks on to

Further author information: (Send correspondence to T.M.: E-mail: tmcinern@scs.ryerson.ca, Telephone: 1 416 979 5000 x7245)

Medical Imaging 2005: Image Processing, edited by J. Michael Fitzpatrick, Joseph M. Reinhardt, Proc. of SPIE Vol. 5747 (SPIE, Bellingham, WA, 2005)
1605-7422/05/$15 · doi: 10.1117/12.594574

the nearby edges, localizing them accurately. Occasionally a Snake will latch onto spurious or neighbor structure boundaries. A user-guided correction step is then required to pull the snake off the incorrect boundaries into the correct position. Furthermore, in noisy regions the user may impose additional constraints, in the form of 'pin' points for example.

An accurate initialization is needed in order for a Snake to lock onto the correct image features. Although Snakes were designed to be intuitively interactive, the goal of rapid and accurate anatomical structure segmentation requires that the user editing stage of the segmentation be minimal. For this reason, Researchers have been actively investigating techniques to mitigate the sensitivity of Snakes to their initialization and making them more automatic. Among these techniques is the use of an inflation force,[4] gradient vector flow fields,[5] and the use of automatic snake element subdivision methods.[6–8] Another approach is to optimize the capabilities of semi-automatic Snakes and other interactive Snake-like algorithms, to the point where only a small amount of time and labor is required to process complex data sets. This approach involves the development of more effective user initialization mechanisms,[2, 9, 10] or control mechanisms that can guide the optimization-driven segmentation process at an appropriately high level of abstraction.[11] Other researchers have focussed on a class of models related to Snakes known as deformable templates. These models are designed to be more specific and automatic by incorporating some form of prior information about object shape and/or object image intensities.[12, 13]

The mathematical formulation of Snakes describes a continuous curve and continuous potential energy functions or forces. To compute a minimum energy solution numerically, it is necessary to discretize the energy functions and to geometrically represent a Snake as a linear combination of basis functions, such as finite elements[2, 4] or geometric splines,[9] or more simply as a set of points connected by edges.[6] Each geometric representation has its own advantages and disadvantages. Advantages of the discrete point-set representation are efficiency, simplicity and increased accuracy for highly curved objects, exhibiting rapid shape variation. However, the increased number of snake points (degrees of freedom) presents problems of control when dealing with noisy data, often necessitating significant (and tedious) user-interaction to produce a correct segmentation. In contrast, finite element- and spline-based Snakes are compact representations (contain fewer degrees of freedom) and and also provide the ability to calculate differential quantities. They are more amenable to higher-level user interaction and control mechanisms in some respects, and are generally more robust against noise. However, in other respects, their increased mathematical and numerical complexity can result in difficulties dynamically adapting their parameterization to suit changing conditions in the image.

3. JESS: JAVA EXTENSIBLE SNAKES SYSTEM

The underlying technology of JESS is Java, a freely available (from *Sun Microsystems*) object-oriented programming language that provides advantages such as modularity, extensibility, multiple threads of control, dynamic loading, portability, and a huge set of standard library classes. Java was designed to be simple and architecture-neutral. When coupled with the object-oriented paradigm, the language allows for rapid and robust software design.

Java also contains a vast set of library classes. JESS makes use of the Java Foundation Classes (JFC), which include *Swing*. These are a set of Java class libraries that support building graphical user interfaces (GUI's) and graphics functionality for applications. They will run on any platform with uniform behavior. Furthermore, several integrated software development programs, such as the freely available *NetBeans* from Sun, allow for the interactive construction and layout of GUI's.

JESS also makes use of the new *Java Advanced Imaging (JAI)* API. JAI further extends the Java platform by allowing sophisticated, high-performance image processing to be incorporated into Java applications. JAI implements a set of core image processing capabilities including image tiling, regions of interest, and deferred execution. JAI also offers a set of image processing operators including many common point, area, and frequency-domain operators. The API is highly extensible, allowing new image processing operations to be added in such a way as to appear to be a native part of it. JAI unifies the notions of image and operator by making both subclasses of a common parent. An operator object is instantiated with one or more image sources and other parameters. This operator object may then become an image source for the next operator object. The connections between the objects define the flow of processed data. The resulting editable graphs of image processing operations may be defined and instantiated as needed.

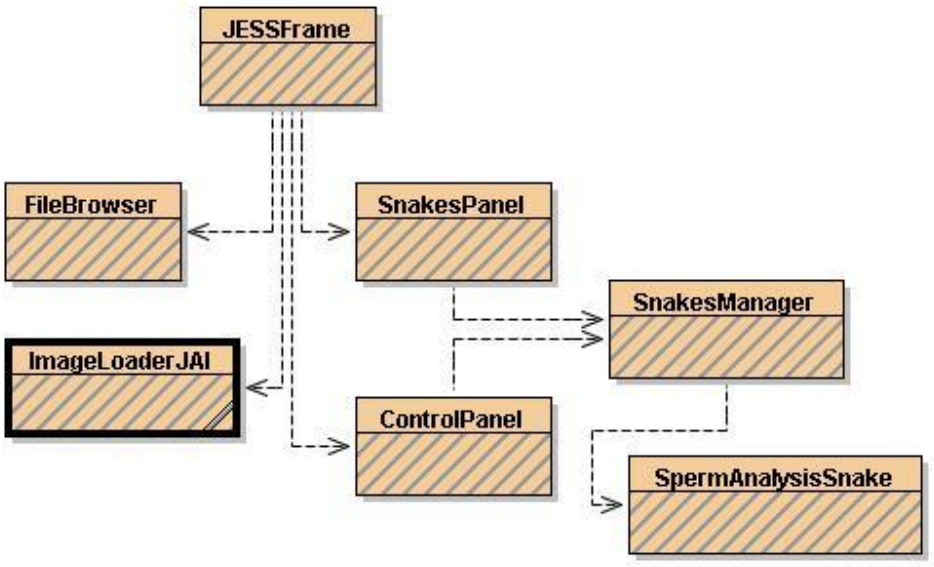


Figure 1. Main class structure of JESS.

A simplified view of the class structure of JESS is depicted in Figure 1. The *JESSFrame* class is a subclass of a Swing *JFrame*. It contains the *main* method and is responsible for setting up and initializing the system. The *JESSFrame* class creates a Snakes control panel, a file browser panel, a JAI image loader object (used by the file browser to load input images of various formats), and a Snakes image panel. The Snakes control panel interacts with the Snakes Manager class (via the Snakes image panel) and provides user controls to set Snakes parameters and control Snake execution. The Snakes panel is a subclass of a Swing *JPanel*. It sets up and controls the preprocessing (via JAI) and drawing of the input image and all Snakes, user interaction with a selected Snake (via mouse event handlers), the Snakes manager object, and the creation/execution of a Snake (via the Snakes manager). Once the basic structure of JESS is understood, the panels set up by *JESSFrame* can be quickly modified or extended to suit a particular application.

3.1. Snakes in JESS

JESS implements the original Snakes algorithm in a hierarchical manner (Fig. 2). At the top of the hierarchy is an abstract Snake class defining functionality common to all Snakes. A Snake is essentially defined as an ordered list of points (implicitly connected by edges). Standard methods to draw, update, and compute forces on, a Snake are all included. Several major subclasses have been implemented, including the original Kass, Witkin, Terzopoulos snake *KWTSnake*, and a new, powerful class of Snakes that is based on subdivision curve theory.[14]

The underlying idea behind subdivision curves is the use of geometric algorithms to progressively subdivide a control polygon. One such algorithm is based upon corner cutting where the algorithm generates a new control polygon by cutting the corners off the original one. Figure 3 illustrates this idea, where an initial control polygon has been refined into a second polygon (slightly offset) by cutting off the corners of the first sequence. The corners of the second control polygon are then cut off, producing a third sequence, etc. In the limit, a smooth curve is generated. Typically only 2 or 3 subdivisions are required. A Snake based on such a representation typically uses the coarsest level control polygon as the Snake degrees of freedom and uses the finest level as "sensors". Forces are computed at the sensor points and then distributed, using weights derived from the original subdivision rules, to the control points. This provides a robust, easily manipulated, always smooth, low degree of freedom Snake that makes use of all available image pixels between control points. A subdivision curve Snake is thus as robust against noise as a finite element-based or B-spline based snake but is much easier to manipulate, like a discrete point-set based Snake.

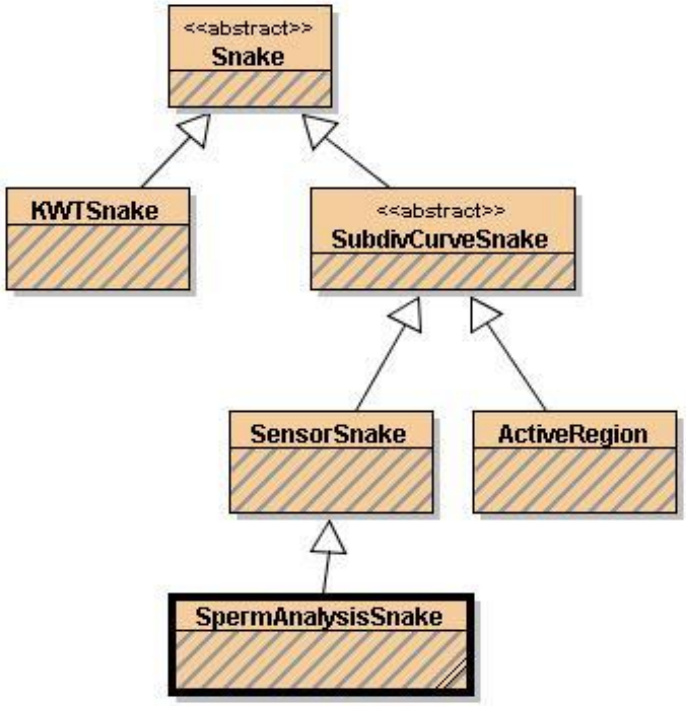
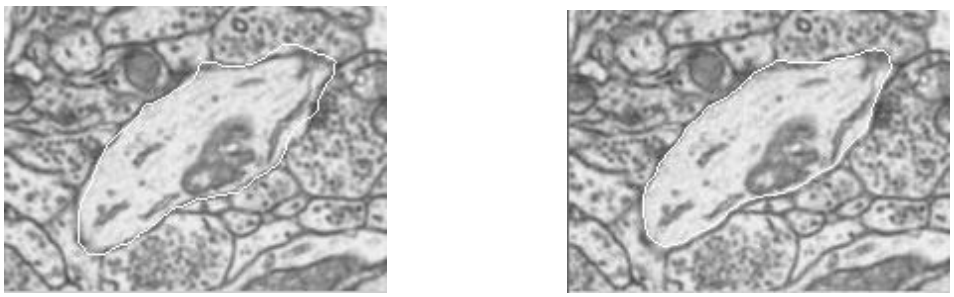


Figure 2. Snakes class hierarchy in JESS.

(a) (b) (c)

Figure 3. Example of the corner cutting subdivision process to used to generate a subdivision curve. (a) The initial control polygon. (b) After one subdivision. (c) The control polygon and curve after two subdivisions. Note how the curve is becoming more smooth after each subdivision.



(a) (b)

Figure 4. Segmenting a neuronal cell with a classical Kass, Witkin, Terzopoulos (*KWTSnake*). The user has drawn a rough contour in (a) and the Snake quickly locks on to the cell boundary in (b).

Two subclasses of the subdivision curve Snake are supported in JESS. The first, which we call a *SensorSnake*, is a closed or open Snake that searches for image features along directions normal to the sensor points. If a feature point is found, a strong spring force is used to pull the sensor point towards the image feature point. The image feature type and search range are, of course, customizable. The second Snake, called an *Active Region*, is a closed curve with the capability of local subdivision. Forces are computed similarly to the Sensor Snake and the control polygon edges are subdivided when they reach a threshold length, creating a Snake that behaves as a sophisticated, robust region growing algorithm.

JESS provides several types of Snakes initialization and interaction mechanisms. A snake may be initialized by drawing a rough contour close to the boundary of the target object. Alternatively, a single point may be entered with the mouse and a circular snake of a user-specified radius and number of points will be created. The user may also click a number of points to form a closed or open control polygon. This polygon is then automatically subdivided to form a smooth curve. Finally, the user may draw a series of lines across an object.[9] These lines are then used to form a closed control polygon and again, a smooth subdivision curve is constructed.

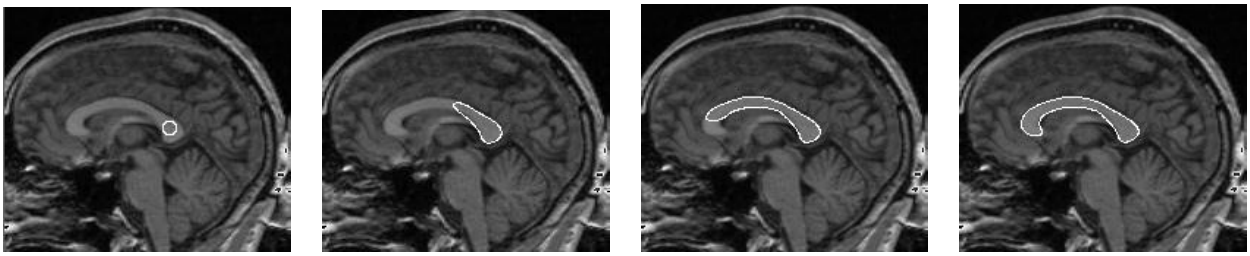
While a Snake is deforming, the user may use the mouse and pull on the Snake, or create 'pin' points that attract the closest point on the Snake towards the pin point. Alternatively, if a subdivision curve Snake was created, the user may stop the Snake deformation and directly edit the position of the control points, automatically affecting the position of a curve segment.

In summary, these highly extensible Snakes classes, along with the user initialization and interaction mechanisms, provide the flexibility and power needed to quickly design custom segmentation systems.

4. RESULTS

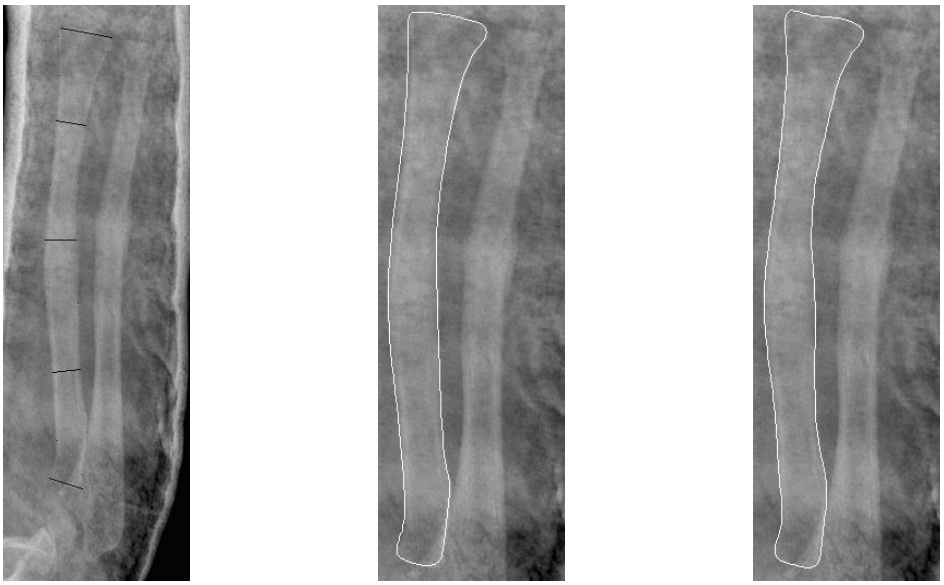
In this section we demonstrate some of the capabilities of JESS. We first demonstrate the use of the various Snakes classes for segmenting anatomical and cellular structures from biomedical images. In Figure 4, a standard *KWTSnake* is used to segment a neuronal cell from an EM photomicrograph. The Snake is initialized by drawing a rough curve around the cell boundary. In Figure 5, an ActiveRegion is initialized with a single mouse click inside the corpus callosum of an MR brain image slice. The ActiveRegion Snake is pulled towards the edges of the corpus callosum (CC) and its control polygon is automatically subdivided. A new subdivision curve is then generated and the process repeats until the entire CC has been segmented. Finally, in Figure 6 the control polygon and smooth limit curve of a Sensor Snake are generated from lines input by the user. This example shows the power of a subdivision curve-based Snake as the initial Snake is almost the exact shape of the target arm bone in this very noisy X-ray image. Like any local optimization algorithm, a Snake will perform best (i.e. efficiently, robustly, accurately) when it's initialization position and shape are close to the final position and shape.

We have also used JESS to implement a prototype semi-automatic sperm analysis system (Fig. 7). The requirements of the system were the fast and accurate, user-controllable segmentation and subsequent shape analysis of human sperm cells from microscopy images. The images required pre-filtering for noise removal and



(a) (b) (c) (d)

Figure 5. Segmenting the Corpus Callosum (CC) from an MR brain image slice. An initially circular *ActiveRegion* Snake is initialized with a single mouse click. The Snake then automatically subdivides and grows along the CC until it is completely segmented.



(a) (b) (c)

Figure 6. Segmenting an arm bone from a noisy X-ray image. (a) A *SensorSnake* is initialized by drawing a series of lines across the object. (b) Slightly magnified view. Note how the initial subdivision curve constructed from these lines is almost the exact shape of the arm bone. (c) The final segmentation. The Snake hardly has to deform to lock onto the correct boundary.

Figure 7. Snapshot of the main panel of a prototype Sperm Analysis System.

edge detection. JAI was used to perform this task. A custom GUI was quickly designed and implemented using the Swing library classes and the GUI generator functionality of NetBeans. A subclass of the Sensor Snake, known as a *SpermAnalysisSnake* (Fig. 2) was developed to perform the segmentation. This essentially only required the setting of a few parameters. A user clicks a single point inside a sperm head and the Snake segments it. Occasionally a minor correction is needed (using a single pin point for example) to prevent the Snake from leaking into the sperm mid-piece. An image containing 20 to 30 sperm cells takes under a minute to process (i.e. limited by how fast the user can move the mouse from sperm cell to sperm cell - the segmentation of a cell itself is instantaneous). Once the sample is segmented, each sperm head is represented by a smooth subdivision curve and any sort of morphology analysis can easily be performed.

5. CONCLUSIONS

This paper describes JESS - a Java-based Extensible Snakes System. This highly extensible, portable, and open-source system combines Java Foundation Classes, Java Advanced Imaging, and powerful Snakes classes to allow for the rapid and simple construction of custom biomedical image analysis systems. We have demonstrated the use of JESS for several medical image segmentation tasks, as well as described a prototype semi-automatic sperm analysis system that was constructed using JESS. Future plans involve extending the Snakes hierarchy in JESS to include a topology adaptive subdivision curve Snake. We plan on releasing JESS to the public in early 2005.

6. ACKNOWLEDGMENTS

The sperm images were provided courtesy of Dr. Brendan Mullen, Mt. Sinai Hospital, Toronto, ON, Canada. The arm bone X-ray image is courtesy of Dr. Paul Babyn, Hospital for Sick Children, Toronto, ON, Canada. TM is funded by the Natural Sciences and Engineering Research Council of Canada.

REFERENCES

1. M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," **1**(4), pp. 321–331, 1988.
2. J. Liang, T. McInerney, and D. Terzopoulos, "United snakes," in *Proc. Seventh International Conf. on Computer Vision (ICCV'99)*, (Kerkyra (Corfu), Greece), September 1999.
3. T. McInerney and D. Terzopoulos, "Deformable models in medical image analysis: A survey," *Medical Image Analysis* **1**(2), pp. 91–108, 1996.
4. L. Cohen and I. Cohen, "Finite element methods for active contour models and balloons for 2D and 3D images," **15**, pp. 1131–1147, November 1993.
5. C. Xu and J. L. Prince, "Snakes, shapes, and gradient vector flow," *IEEE Transactions on Image Processing* **7**(3), pp. 359–369, 1998.
6. S. Lobregt and M. Viergever, "A discrete dynamic contour model," **14**, pp. 12–24, March 1995.
7. T. McInerney and D. Terzopoulos, "T-snakes: Topology adaptive snakes," *Medical Image Analysis* **4**, pp. 73–91, 2000.
8. V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," pp. 694–699, IEEE Computer Society Press, (Los Alamitos, CA), 1995.
9. T. McInerney and H. Dehmeshki, "User-defined B-spline template snakes," in *Proc. Sixth International Conf. on Medical Image Computing and Computer-Assisted Intervention (MICCAI 2003)*, pp. 746–753, Springer, (Montreal, Canada), November 2003.
10. E. N. Mortensen and W. A. Barrett, "Interactive segmentation with intelligent scissors," *Graphical Models and Image Processing* **60**, pp. 349–384, 1998.
11. T. McInerney, G. Hamarneh, and D. Terzopoulos, "Deformable organisms for automatic medical image analysis," *Medical Image Analysis* **6**, pp. 251–266, 2002.
12. L. Staib and J. Duncan, "Boundary finding with parametrically deformable models," **14**, pp. 1061–1075, November 1992.
13. T. Cootes, A. Hill, C. Taylor, and J. Haslam, "The use of active shape models for locating structures in medical images," **12**, pp. 355–366, July 1994.
14. G. Chaikin, "An algorithm for high speed curve generation," *Computer Graphics and Image Processing* **3**, pp. 346–349, 1974.