Implementation of B-snakes

Introduction:

Active contours or Snakes are used heavily in computer vision for boundary delineation or edge detection. A **snake** is defined as an energy minimizing spline - whose energy depends on its shape and location within the image. Shape of the snake is controlled by the internal forces (the implicit model) and external forces (image data). Internal force acts as smoothing constraint for the snake and the external force guide the snake towards the features in the image.

Let $v(s) = (x(s), y(s))$ is the parametric representation of the snake where the value of s is within the range $(0,1)$. Then total energy of the snake can be represented as –

$$E_{snake} = \int_0^1 E_{snake}(\mathbf{v}(s))\, ds$$

$$= \int_0^1 E_{int}(\mathbf{v}(s)) + E_{ext}(\mathbf{v}(s))$$

Where E_{int} is the internal energy of the snake and E_{ext} is the external energy of the snake. Here the objective is to find such $v(s)$ so that the total energy of the snake is minimized. Minimization of the above equations involves calculus of variation and solving Eular equation. For discrete case we get the following equations –

$$\mathbf{Ax} + \mathbf{f}_x(\mathbf{x}, \mathbf{y}) = 0$$

$$\mathbf{Ay} + \mathbf{f}_y(\mathbf{x}, \mathbf{y}) = 0$$

In order to evolve the snake, the expression of the snake can be given as a function time by –

$$\mathbf{x}_t = (\mathbf{A} + \gamma\mathbf{I})^{-1}(\gamma\mathbf{x}_{t-1} - \mathbf{f}_x(x_{t-1}, y_{t-1}))$$

$$\mathbf{y}_t = (\mathbf{A} + \gamma\mathbf{I})^{-1}(\gamma\mathbf{y}_{t-1} - \mathbf{f}_y(x_{t-1}, y_{t-1}))$$

In this project I have implemented B-spline representation of the snake. This B-spline representation of snake is called B-snake. B-snake is represented by a linear combination of B-spline basis functions and a set of control vertices $V_i = (X_i, Y_i)$.

B-snake representation is compact, exhibits local control, and it is possible to include corners by allowing multiple knots. I have used uniform cubic spline basis functions to represent the B-snake.

Formulation of the problem and the algorithm:

Now I will describe how B-snake is implemented in this project. This involves two steps which are given below -

Step1: initialization of the B-snake

We assume that there are (p+1) points on the snake curve and we use (m+1) control points to denote the snake. Here m < p. From the given (p+1) points we want to find an approximate B-spline such that the distance between the curve and the spline approximation is minimized. This distance equation is given by the following equation -

$$R = \sum_{j=0}^{p} | Q(u_j) - P_j |^2 = \sum_{j=0}^{p} ((x(u_j) - x_j)^2 + (y(u_j) - y_j)^2)$$

As the above equation is quadratic equation, the minimum value of R will at points X_i and Y_i such that the following derivatives are zero.

$$\begin{cases} \frac{\partial R}{\partial X_l} = 0 \\ \frac{\partial R}{\partial Y_l} = 0 \end{cases}$$

By solving the derivative we get the following system of linear equations in X_i and Y_i where x is from 0 to l.

$$\sum_{i=0}^{m} X_i \sum_{j=0}^{p} B_i(u_j) B_l(u_j) = \sum_{j=0}^{p} x_j B_l(u_j)$$

$$\sum_{i=0}^{m} Y_i \sum_{j=0}^{p} B_i(u_j) B_l(u_j) = \sum_{j=0}^{p} y_j B_l(u_j)$$

We solve the above system of linear equation and get the initial B-snake.

Step 2: - Minimization of the snake energy

In this step we minimize the total energy of the spline curve. Total energy of the spline curve is given by ---

$$E = \sum_{j=0}^{p} \{\frac{1}{2}\alpha(u_j)[(\sum_{i=0}^{m} X_i B_i'(u_j))^2 + (\sum_{i=0}^{m} Y_i B_i'(u_j))^2]$$

$$+ \frac{1}{2}\beta(u_j)[(\sum_{i=0}^{m} X_i B_i''(u_j))^2 + (\sum_{i=0}^{m} Y_i B_i''(u_j))^2]$$

$$+ F(v(u_j))\}$$

Here we want to find such control points which minimize the energy of the curve by satisfying -

$$\forall l \in \{0, \cdots, m\} \begin{cases} \frac{\partial E}{\partial X_l} = 0 \\ \frac{\partial E}{\partial Y_l} = 0 \end{cases}$$

This yields the following system of linear equations for X_i s-

$$\sum_{i=0}^{m} X_i \ [\sum_{j=0}^{p} \alpha(u_j)B_l'(u_j)B_i'(u_j) + \sum_{j=0}^{p} \beta(u_j)B_l''(u_j)B_i''(u_j)]$$
$$+ \ \sum_{j=0}^{p} B_l(u_j)F_x(v(u_j)) = 0$$

This system of linear equation can be solved iteratively by the way similar to the original snake algorithms.

$$\begin{cases} X_{t+1} &= (A_b + \gamma I)^{-1}(\gamma X_t - G_x(x_t, y_t)) \\ Y_{t+1} &= (A_b + \gamma I)^{-1}(\gamma Y_t - G_y(x_t, y_t)) \end{cases}$$

Thus we get a new set of control points. By that minimize the total energy of the B-spline approximation of the snake. When the total energy is minimized we get our final snake. We stop the iteration when the total energy of the snake at two successive steps is same.

Implementation:

I have implemented the project in MATLAB. The program at first take inputs from user. The inputs include name of the image file, the parameters alpha, beta, gamma and number of control points to be used to approximate the snake. Then it initializes the snake by forming a B-spline approximation of the snake curve. After that the control points are evolved and the snake finally goes to its desired location where its energy is minimum.

Results:

Output of my program justifies that the B-spline snake deforms to find the boundary in the image and evolves. Figures 1 to 6 in the next page show how B-snake finds the image boundary in the images.
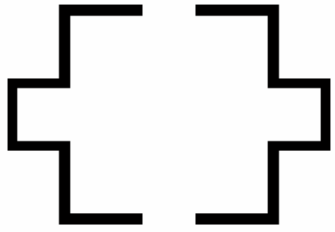
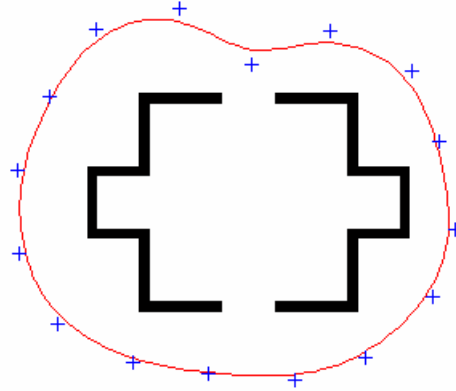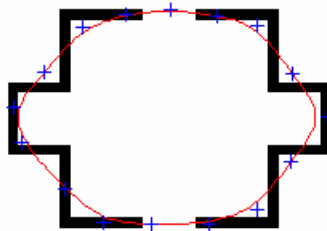Fig1. Image room.pgm.   Fig2. Initial B-snake with control points

Fig 3. Final B-spline of minimum energy
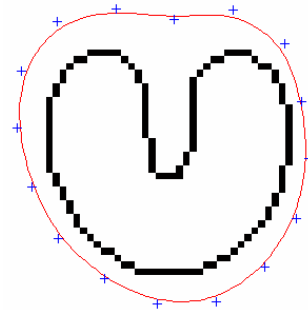
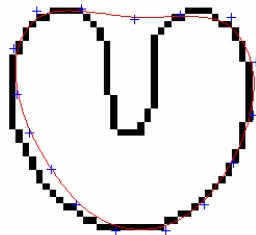Fig 4. Image U64.pgm   Fig 5. Initial B-snake with control points

Fig 6. Final B-snake of minimum energy

Discussion:

For B-spline snake we use the control points only to evolve the snake. That significantly reduces the computation time of calculating the stiffness matrix. Numbers of control points that determine the shape of the snake have influence of the performance of the algorithm. If we use few points then the snake does not evolve well. Here the step size gamma is large with respect to the value used in normal snake algorithm. Moreover B-snake converges quickly then the traditional snake algorithm. By using non-uniform B-spline and allowing multiple knots, we can create the snake which can find corners in the image.

Conclusion

B snake is a good approximation of the traditional snake. B snake represent the snake compactly, allow corners, and converge fast. Thus it can be applied instead of snake where we need efficient implementation.

Reference:

1. G. Medioni, S. Menet and P. Saint-Marc, "B-Snakes: Implementation and Application to Stereo", Proceedings of the 7th Israeli Conference on Artificial Intelligence, Vision and Pattern Recognition, pp. 223-236, RamatGan, Israel, December 1990

2. M. Kass, A. Witkin, and D. Terzopolous, "Snakes: Active contour Models", International Journal of Computer vision", 1988.

3. Andrew Blake, Michel Isard, "Active Contours: the Application of Techniques from Graphics, Control theory and Statistics to Visual Tracking of Shapes in Motion", Springer, 1998.

4. Richard H. Bartels, John C. Beatty, Brian A. Barsky, "An Introduction to Splines for use in Computer Graphics and Geometric Modeling", Morgan Kaufmann Publishers inc, 1987

Prepared By:

S. M. Shahed Nejhum.
smshahed@cise.ufl.edu
UF-ID – 85891199
Ph.D. Student,
CISE Department,
University of Florida.