CSUMS First Semester

Chris Cacciatore
Department of Mathematics
Student – University of Massachusetts Dartmouth

Advisors: Saeja Kim and Sigal Gottlieb
Department of Mathematics
Professors – Unversity Massachusetts Dartmouth

May 11, 2011

Abstract

This project focuses on the use of Radial Basis Functions (RBF's) in edge detection in both one-dimensional and two-dimensional images. We will be using a 2-D iterative edge detection method. We will be varying the point distribution and shape parameter. We will also quantify the effects of the accuracy of the edge detection on 2-D images. Furthermore, we will study a variety of different RBF's and their accuracy in edge detection.

1 Introduction

Coming into CSUMS this semester both my partners and I have had little experience in Mathematical research as well as any type of computer programming. After familiarizing ourselves with the coding enviroment MatLab, Professor Gottlieb assigned us with a project in applying RBF's to edge detection. She provided us with a code her and her colleague wrote which initially used Multi-qudric RBF's to operate.

$$\phi = \sqrt{(x - x_i)^2 + \epsilon_i^2}$$

It was our task to understand this code and to be able to alter it in order to recieve the best possible results.

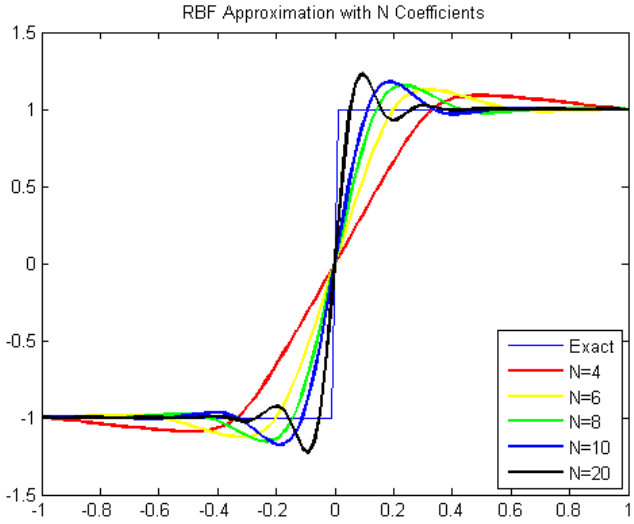
2 Problem

The initial problem we adressed this semester was to work our way around the program TwoD Example1 and learn how certain pieces of the code changed the results. At first by simply adjusting the value of the shape paramter epsilon. Since then, we have worked with different types of RBF's other than the Multi-quadric which was given to us initially. We needed to develop lines of code for the Inverse Multi-quadric and Gaussian RBF's as well in order to use them in the creation of edge maps. We hoped there would be some difference in the way these three functions mapped out the edges of the given image. After this had been accomplished we left our research to answer questions that may have been brought about by our classmates. Initial coding for MQ RBF:

```
%%%%%%%%%%%%%%%%%% this code is for multi-quadric RBF
M(ix,iy) = sqrt( (x(ix)-x(iy))^2 + (eps(iy))^2);
if M(ix,iy) == 0
 MD(ix,iy) = 0;
 else
 MD(ix,iy) = (x(ix) - x(iy))/M(ix,iy);
 end
```

3 MQ RBF

Though this is a piece of our project we saved for last, it is something that should be stated earlier. Our project involves the use of a Multi-quadric RBF which is used in the code as an approximator. It uses a sumation of MQ RBF's with a certain value of epsilon evaluated anywhere from 1 to N coefficients. In one case we used this sumation to approximate a common step function beginning with 4 coefficients then increasing until we reach 20.



As we continue to increase the number of coefficients our approximation begins to look more and more similar to the actual function we are approximating aside from the outgoing spikes along the vertical line of the step function. This is known as the Gibbs Phenomenon. It is each time this

phenomenon occurs that the code TwoD Example1 draws out edges for the image being mapped.

4 Technique

The techniques used to go about accomplishing our initial task was really just an educated guess-and-check method. We began by choosing any arbitrary number for epsilon, then we either increased or decreased the origianl number in accordance with the accuracy of the edge map being produced, though accuracy could only be determined by eye as opposed to an actual mathematical method of determining which edge maps were better. For example the edge map we had determined was the best while using a Multi-quadric RBF was at epsilon = .1.

Since we have come to this conclusion we have learned how to put both the Inverse Multi-quadric RBF:

$$\phi = \frac{1}{\sqrt{(x - x_i)^2 + \epsilon_i^2}}$$

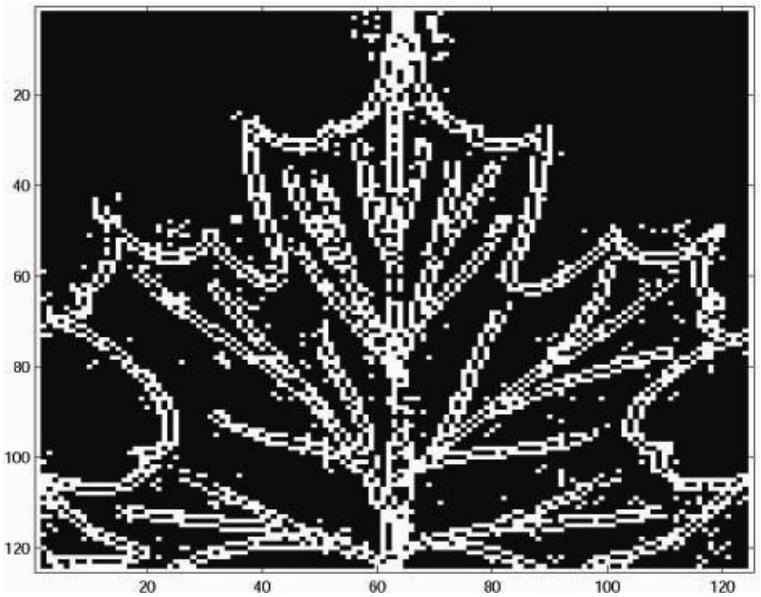
and the Gaussian RBF:

$$\phi = e^{(-\epsilon^2 * (x - x_o)^2)}$$

into our code. After some testing with different values of epsilon for the Inverse Multi-quadric and Gaussian RBF's we determined that the code:

```
 M(ix,iy) = 1/sqrt( (x(ix)-x(iy))^2 + (eps(iy))^2);
if M(ix,iy) == 0
  MD(ix,iy) = 0;
else
MD(ix,iy) = -(x(ix) - x(iy))/sqrt( ((x(ix)-x(iy))^2 + (eps(iy))^2)^3);
end
```

with an epsilon value of .69 was the most accurate value for the InvMQ RBF:



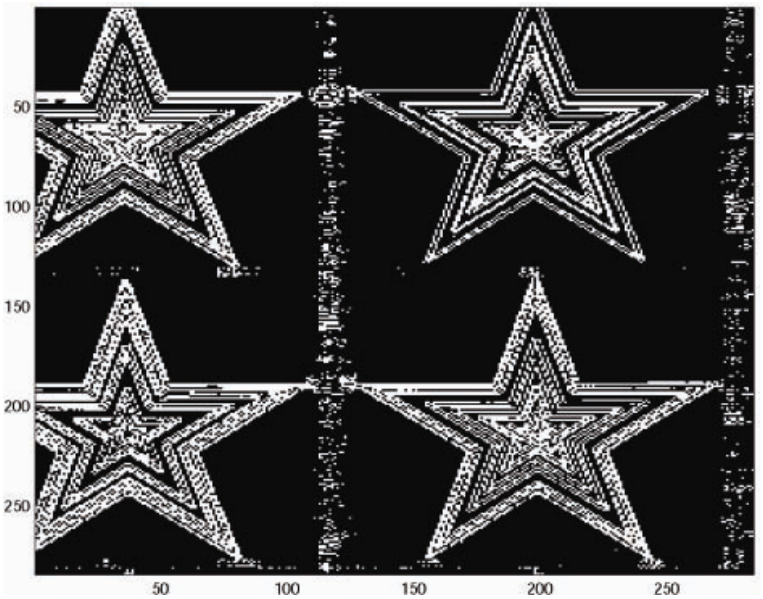
and we have determined that epsilon at 3 gives us the best edge map for the Gaussian RBF:

using the adjusted code:

```
M(ix,iy) = exp(-((eps(iy))^2)*((x(ix)-x(iy))^2));
if M(ix,iy) == 0
   MD(ix,iy) = 0;
else
MD(ix,iy) = -2*((eps(iy))^2)*(x(ix)-x(iy))*exp(-((eps(iy))^2)*(x(ix)-x(iy))
end
```

5 Progress

So far our group has completed the tasks asked of us, although there are some things I would personally like to have worked more with. I admit that I still do not completely understand why our code for edge detection though for the most part I do understand how it works. I am aware of the involvement of the lambda matrix though am not in full understanding. I was however, able to come to an answer to a question asked during my second in-class presentation by Zack Grant. He asked if different values of epsilon would be required to create edge maps of images of greater size. For example, the maple leaf image used throughout this report has dimensions of 125 x 125. So i went and used an image with dimensions of 284 x 284:



There is some noise in the edge map though the main focus of the edge map, the stars themselves, hold a close accuracy to those of the original image. This is only one example, however it seems as though the answer to the question is no, the same values of epsilon render edge maps that are just as accurate for larger images as they are for smaller ones. However, the run time did jump from about 25 seconds to about 5 minutes. A major inconvienence if one were to be using these programs for any sort of business reasons. Maybe some future work on this project could be to cut down on run time and/or noise in the edge maps.

6 CSUMS

I enjoyed my first semester of CSUMS and felt it really prepared me to take on more difficult research projects in the future. It was good to be able and do some work in the field I plan on going into after my college graduation. Up until now math for me has just been solving problems using calculus and trig functions. With CSUMS I have been able to apply my math skills to a specific job as opposed to just thinking, "I will need this skill someday".

Positives:

1. I enjoyed many of the presentaions given during class. Although some of

them were far out of my league, it introduced me to many different topics I had never even thought of.

2. The undergraduate Research Conference at Umass Amherst was a good experience for me. I have never done anything like that before and i enjoyed being able to look at what so many people have been working on.

Difficulties:

1. I did have trouble with a lot of the technical material towards the beginning to the project. I had no experience in computer programming and am usually pretty bad with electronics in general.

7 References

1. Vincent Durante, Jae-Hun Jung. An iterative adaptive multi-quadric radial basis function method for the detection of local jump discontinuities. Appl. Numer. Math. 57 (2007) 213-229