Acknowledgments

Acknowledgements:-

At the beginning we would like to thank our supervisor Dr. Jamal Zemerly for helping us with our project and being there when we seeked his assistance.

We also would like to thank Dr. Halim for allowing us to use his picture to test our edge detection operators.

Finally, we would like to direct our thanks to abdulrahman al taboor who gave us some helpful hints that helped us finishing our project and presenting you with our final product.

At the end, we would like to thank you, readers, for giving us some of your time by reading our report; we hope that you'll enjoy it.

Chapter 1

Introduction

1) Introduction:-

1.1 Goals and Objectives:-

The main goal of our project is to detect any edges that are found in any greyscale image that the user inputs into the program after applying one of the five edge detection operators that we have coded in our program.

A text user interface appears after the user operates our program when the image that he wants to use is chosen. The menu includes some tips and helpful information that the user could use, it also asks the user to choose one of the five edge detection operators. After choosing the type of operator the user is asked if he wants to display the "thresholded "image or display "the edge detected image". At the end, the user is asked if he wants to superimpose the edges on the original image.

Finally the program will ask the user if he wants to change the edge detection operator and use it on the same image he chose at the beginning, or if he wanted to exit the program

1.2 Overview of the report:-

In *Chapter two*, the system requirements will be shown, which will include information about the system requirements and specifications, functional specifications, inputs, outputs, operators and how they work and the limitations of our program.

Chapter Three will deal with the implementation of our program, which will be divided into the input, output, the different operators and the XV program that we have used.

Chapter Four will talk about the different tests that we ran on our program to make sure that its working correctly.

Chapter Five will include the critical appraisal, applications and future recommendations.

A reference page and an appendix which will include the source code of our program will be attached to the end of our report.

Chapter 2

System Overview

2) System Overview:-

2.1 Requirements:-

UNIX or Linux operating systems must be used to operate the program. XV must be installed to be called by the program to display the images. The input image must be in Grayscale PGM ASCII image format.

2.2 System Specifications:-

The program was written in C++ programming language.

2.2.1 Functional Specifications:

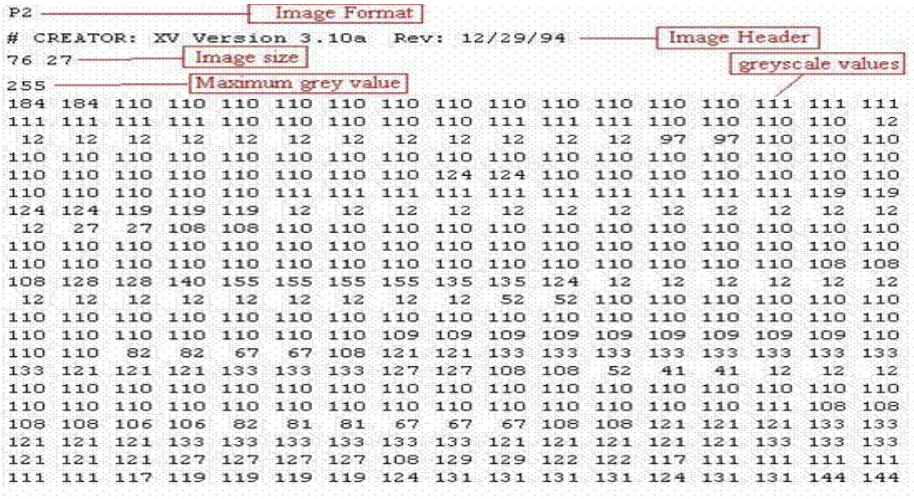
The main functions in our program are :

- Reading a grayscale ASCII image.
- Applying one of the following operators:-
    1- Sobel operator
    2- Prewitt operator
    3- Robert operator
    4- Gradient operator
    5- Kirsch operator
- Ask the user if he wants to threshold the image.
- Display the output using the XV program

2.2.2 Inputs:

The main input in our program is:

- A grayscale PGM ASCII image format, where we can store the image size in an array after reading the input image size which is found on the third line, which could be seen in fig .1 below.



(Fig.1)

2.2.3 Outputs:

Our output will be:

- The original image will be displayed by calling the xv program (software used in UNIX or Linux to display images).
- The edge detected image for the selected operator.
- Thresholded edge detected image using a threshold value that will be entered by the user manually.
- Angled image (the image which results from calculating the angles)
- Superimposed image (the image which results from superimposing the edge detected image over the original image

2.2.4 Operators and how they work:

The edges operators that were used in our program are:

1) Sobel Operator
2) Prewitt Operator
3) Robert Operator
4) Gradient Operator
5) Kirsch Operator

The algorithms for these operators are the following:

Apply one of the five operators
Convolve two types of masks for each operator over the input image
Get the values of the following terms:

- dx value (the value of the horizontal edge)
- dy value (the value of the vertical edge)

Calculate the magnitude and the angle for each pixel of the input image.

2.2.5 Limitations and Restrictions

The following points are the limitations and restrictions of the implemented system:

- The quality of the output either the edge detected image or the thresholding image is not 100% since some of it has unwanted edges (noise).
- Certain images do not give an appropriate output as expected.
- If the input image is very large the program may take some time to display the output and sometimes does not display it at all.
- The program finds the edges for grayscale images only (PGM format).
- Simple text user interface to communicate with the user

Chapter 3

Implementation

3) Implementation:-

3.1 Input and Output:-

The XV program is used to read the greyscale image (PGM ASCII image format). The XV program is also used to display the edge detected output, the threshold output and the superimpose operation's output

3.2 Sobel Operator:-

The Sobel Operator consists of two [3 x 3] masks, one for each axis.

-1	0	1
-2	0	2
-1	0	1

x axis

1	2	1
0	0	0
-1	-2	-1

y axis

(Fig. 2)

3.3 Prewitt Operator:-

The Prewitt operator also consists of two [3 x 3] masks.

-1	0	1
-1	0	1
-1	0	1

x axis

1	1	1
0	0	0
-1	-1	-1

y axis

(Fig. 3)

3.4 Robert Operator:-

The Robert Operator includes two [2 x 2] masks one for the x axis and the other of the y axis.

1	0
0	-1

x axis

0	1
-1	0

y axis

(Fig. 4)

3.5 Gradient Operator:-

The gradient operator also includes two [2 x 2] masks for the x and y axis

-1	1
0	0

x axis

1	0
-1	0

y axis

(Fig. 5)

3.6 Kirsch Operator:-

The Kirsch operator has two [3 x 3] masks

3	3	3
3	0	3
-5	-5	-5

x axis

3	3	3
3	0	3
-5	-5	-5

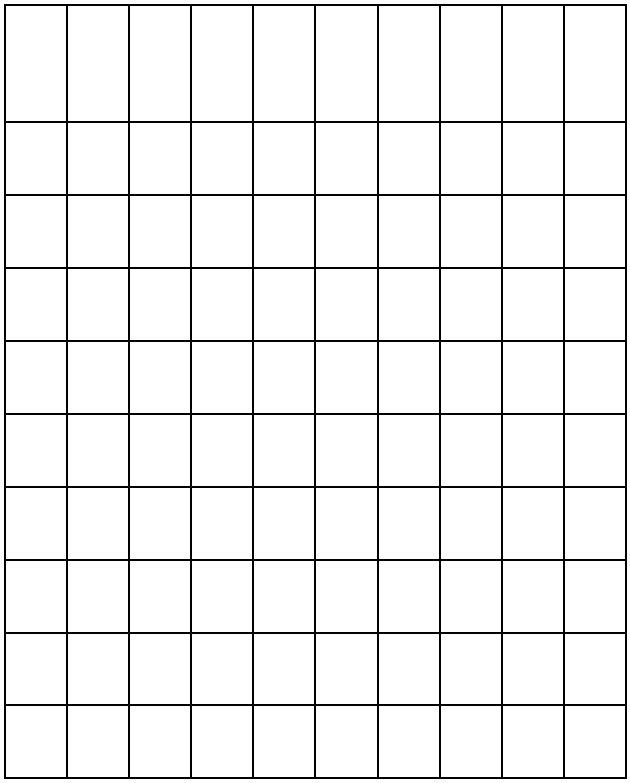
y axis

(Fig. 6)

3.7 How do these operators work?:-

The algorithms of the (Sobel, Prewitt & Kirsch) operators are the same except that they have different mask cell's value while the (Robert & Gradient) operators have the same procedure since they have 2 x 2 masks and yet they differ in getting the value of the horizontal and vertical edges.

The five operators operate in a similar manner, where first of all the input image will be transferred into a 2D array as shown in (fig 7):



(Fig. 7)
The original image

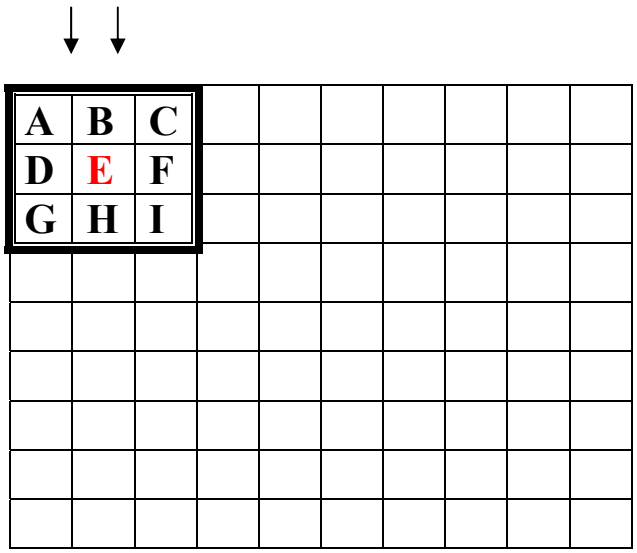
Each element of this array will have a value between 0 and 255 that represents the gray scale value of this PGM image.

3.7.1 Convolution of the Masks:

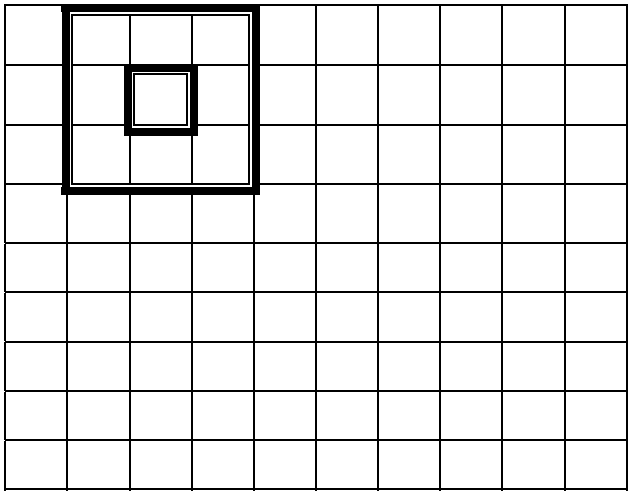
In the case of Sobel, Prewitt or Kirsch operators where all of them have (3 x 3) masks, the convolving process is done by super imposing the two masks individually over the original image's pixels as shown in (Fig. 8) where the pixel under the center cell (5) is the one being affected, then the mask will slide over the next set of pixels as shown in (Fig.9) and so on until it covers the whole image.

.

-1	**0**	**1**
-2	**0**	**2**
-1	**0**	**1**



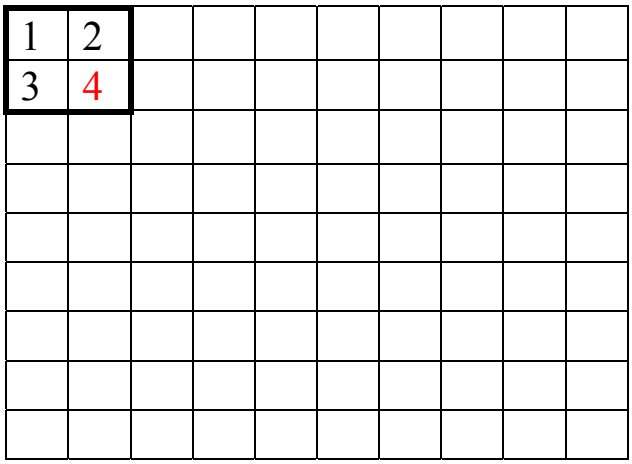
(Fig. 8)



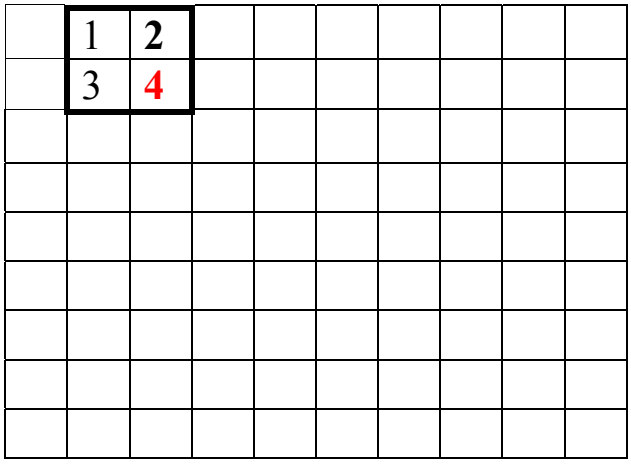
(Fig.9)

In the case of Robert & Gradient operators where all of them have (2 x 2) masks, the convolving process is done by super imposing the two masks individually over the original image's pixels as shown in (Fig. 10) where the pixel (4) is the one being affected in Gradient operator while the pixel (3) is the one being affected in Robert operator, then the mask will slide over the next set of pixels as shown in (Fig.11) and so on until it covers the whole image

1	**0**
0	**-1**



(Fig. 10)



(Fig.11)

3.7.2 Calculating the values of (dx & dy):

Within each convolving process the program will calculate the values of dx & dy (the values of the horizontal and vertical edges) for each affected pixels of the original image, these calculations are done by multiplying all of the pixel values in the original image by the values in the cells over them and then add the results together, for simplifications the following equations are used to find the values of dx and dy for each operator:

In case the operator's mask is 3x3 consider the following area where the mask covers it.

A	**B**	**C**
D	**E**	**F**
G	**H**	**I**

A) *Sobel equations:*

dx = value of pixel A x 1 + value of pixel B x 2 + value of pixel C x 1
+ value of pixel D x 0 + value of pixel E x 0 + value of pixel F x 0
+ value of pixel G x –1 + value of pixel H x –2 + value of pixel I x –1

dy = value of pixel A x –1 + value of pixel B x 0 + value of pixel C x 1
+ value of pixel D x –2 + value of pixel E x 0 + value of pixel F x 2
+ value of pixel G x –1 + value of pixel H x 0 + value of pixel I x 1

B) *Prewitt equations:*

dx = value of pixel A x 1 + value of pixel B x 1 + value of pixel C x 1
+ value of pixel D x 0 + value of pixel E x 0 + value of pixel F x 0
+ value of pixel G x –1 + value of pixel H x –1 + value of pixel I x –1

dy = value of pixel A x –1 + value of pixel B x 0 + value of pixel C x 1
+ value of pixel D x –1 + value of pixel E x 0 + value of pixel F x 1
+ value of pixel G x –1 + value of pixel H x 0 + value of pixel I x 1

C) *Kirsch equations:*

dx = value of pixel A x 3 + value of pixel B x 3 + value of C x 3
+ value of pixel D x 3 + value of pixel E x 0 + value of F x 3
+ value of pixel G x –5 + value of pixel H x –5 + value of I x –5

dy = value of pixel A x –5 + value of pixel B x 3 + value of pixel C x 3
+ value of pixel D x –5 + value of pixel E x 0 + value of pixel F x 3
+ value of pixel G x –5 + value of pixel H x 3 + value of pixel I x 3

In case the operator's mask is 2 x 2 so consider the following area which is covered by the mask.

A	**B**
C	**D**

* The pixel D is the affected pixel in the Gradient operator
* The pixel C is the affected pixel in the Robert operator

A*) Robert equations:*

dx = value of pixel A x 1 + value of pixel B x 0
+ value of pixel C x 0 + value of pixel D x –1

dy = value of pixel A x 0 + value of pixel B x 1
+ value of pixel C x –1 + value of pixel D x 0

B) *Gradient equations*:

dx = value of pixel A x –1 + value of pixel B x 1
+ value of pixel C x 0 + value of pixel D x0

dy = value of pixel A x 1 + value of pixel B x 0
+ value of pixel C x –1 + value of pixel D x 0

3.7.3 Calculating the Magnitude & Angle:

After getting the values of dx & dy for each convolving process the program will calculate the magnitude and the angle for the affected pixel in the original image, the value of the magnitude will represent the gray value pixel which range 0 up to 255, high values of pixel's magnitude in the image correspond to sharp local changes in gray values i.e.(edge).

***The Magnitude is given by:**

$$Magnitude = \sqrt{dx^2 + dy^2}$$

***The Angle is given by:**

$$Angle = \arctan\left(\frac{dy}{dx}\right)$$

3.8 XV program:-

In order to display the image, an external software package, XV, was used. The XV program can display almost all image formats and converts from one format to another and then saves them. It was used in the project to display the original image and the image which was obtained after applying one of the edge detection operators.

Chapter 4

Testing

Testing

4.1 Verification:-

Verification of the implementation can be proved by showing that it satisfies the requirements which are stated in chapter 2. The requirements can be divided into three categories: input image (PGM format), edge operators and output (edge detected, thresholded).

The system was required to find the edges of a grayscale image of a PGM format using different edge operators and display the output which is either the edge detected image or thresholded image.

It appears that the implemented system accepts only PGM format images as an input and then applies one of the five operators that the user has chosen. Finally our program displays the output which is either the edge detected image or the thresholded image.

Some samples are shown in section 4.3

4.2 Validation:-

The validation of the implemented system can be proved by showing that it meets the specification stated in chapter 2. The functional specification of the system was divided into four main functions: reading the input image, applying one of the edge operators, thresholding the edge detected image and displaying the output. The previous four functions were implemented and tested.

The inputs and outputs satisfy what was stated in the specification, except that some images were not giving an appropriate output. The input of the system is a PGM format while the outputs are either the edge detected image or the thresholded image.

Three test scenarios were applied to the implemented system. These scenarios were checking the output for each operator, validating the output of the thresholded image using three values (low, medium and high) and testing the superimposed output.

4.3 Edge detection test:-

The following images have been tested to get the edge detected image using the five implemented operators.

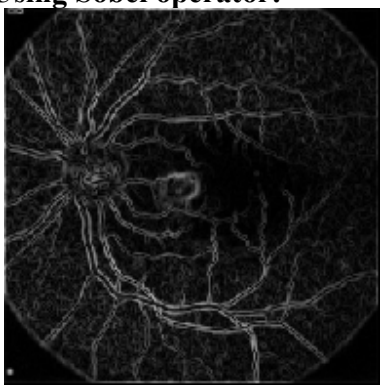
Inputs:

The original PGM image of Eye Retina
- fig 12

Dr. Halim
-fig 13

Outputs:

1) Using Sobel operator:

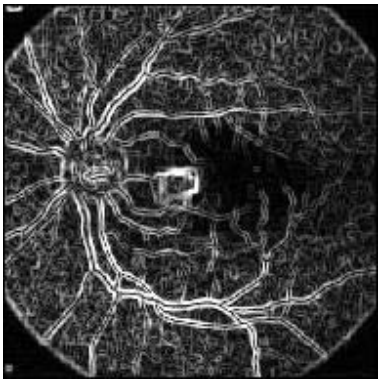


-fig 14

- fig 15

Outputs:

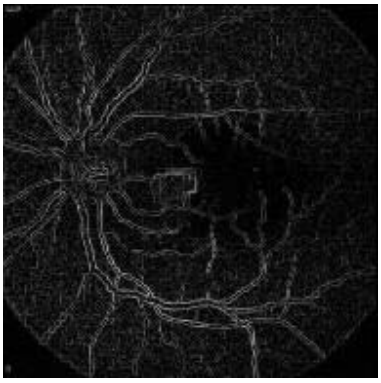
2) Using Prewitt operator:



-fig 16

-fig 17

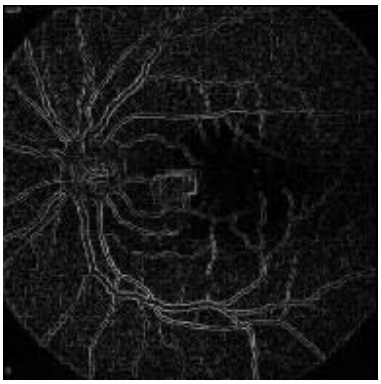
3) Using Robert operator:



- fig 18

-fig 19

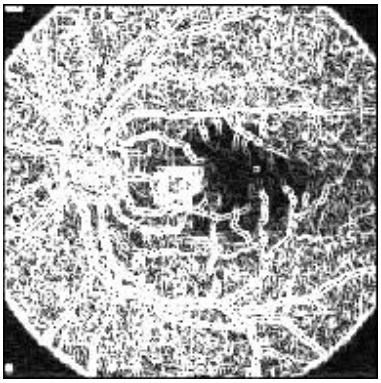
4) Using Gradient operator:



-fig 20

-fig 21

5) Using kirsch operator:



-fig 22

-fig 23

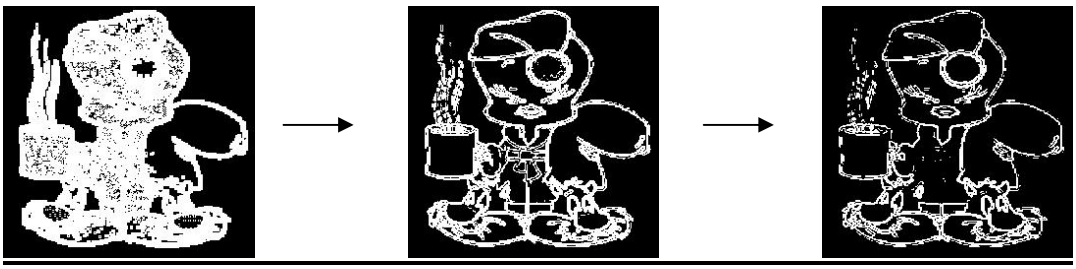
4.4 Thresholding test:-

The following image has been tested using different threshold values (low, medium, high) as shown below:

Inputs:

(fig 24)

Outputs:



Low threshold Medium threshold High threshold

(fig 25)

4.4 Superimposing test:-

The following image has been tested using the superimposing feature:

(fig 26)

Chapter 5

Discussion

Discussion

5.1 Summary:-

In this report, we expressed the idea of our project where the program will detect the edges of a grayscale image of PGM format using different edge detection operators. The requirements and tools needed to run the program, and the system overview that includes information about the system requirements, specifications, functional specifications, inputs, outputs and operators all of them were mentioned at early stages. The implementation of the operator's masks and how their algorithms were applied on the image were discussed in detail.
Test scenarios for different operators were shown using different types of samples. Finally, improvement and recommendations for this project were mentioned in the final stage of this report.

5.2 Critical Appraisal:-

The execution of this project was passed through five stages: input, output, edge detecting, thresholding and testing. A number of problems and difficulties were faced at each stage. The main problem at the input was reading the image where the program will ignore some lines in the image and store the rest. While at the output the main difficulty was displaying the image using xv program. These two problems were solved after some training in using xv and dealing with input /output file processes. In edge detection there were two problems, the first problem was understanding the algorithms of each operator since some of them have a mask of 2x2 cells while the others have a mask of 3x3 cells.

The second problem was the protection in the user interface options, since all the options are numbers and the program will crash if the user enters a character instead of a number, however those two problems were solved after doing some testing using different techniques. The problem with thresholding was in entering the threshold value where the program will crash in case the user inters a float number or a character instead of an integer number. Finally in the testing stage, is has been found that some images after converting them to PGM were not giving the appropriate output as expected. Unfortunately we couldn't solve these two problems due to time limitation.

5.3 Applications:-

There are a lot of applications where edge detection can be used on such as:

- Eye Retina Scanner (i.e. Field of Medicine)
- Digital Art Programs (i.e. Photoshop….etc)
- Landscape Scanner (i.e. Used by geographers)
- Caesar Radar

5.4 Recommendations:-

There are a lot of recommendations that can be applied to this project to make it more effective such as:

- GUI (i.e. Graphical User Interface)
- Auto thresholding using the Mean & Median values
- Thinning (Which is defined as the operation that tides up edge detected images by reducing the lines thickness to one pixel).

References

References

Books Name:	**Authors:**
[1] Photographic imaging techniques in C++	- Craig A.Lindley.
[2] Digital Image Processing	- Rafael C.Gonzalez - Richard E.Woods
[3] C++ how to program	- Deitel& Deitel
[4] Our logbook	-1125, 1133, 1147

Websites:

library.wolfram.com/example/edgedetection

http://www.dai.ed.ac.uk/HIPR2/edgedetct.htm

http://www.ee.bgu.ac.il/~greg/graphics/special.html

http://www.cse.psu.edu/~ferguson/proj2.html

http://www.trilon.com/xv/xv.html

Appendix