

Figure B.6: NG40 Contour Map (Smoothed $\times 40$).

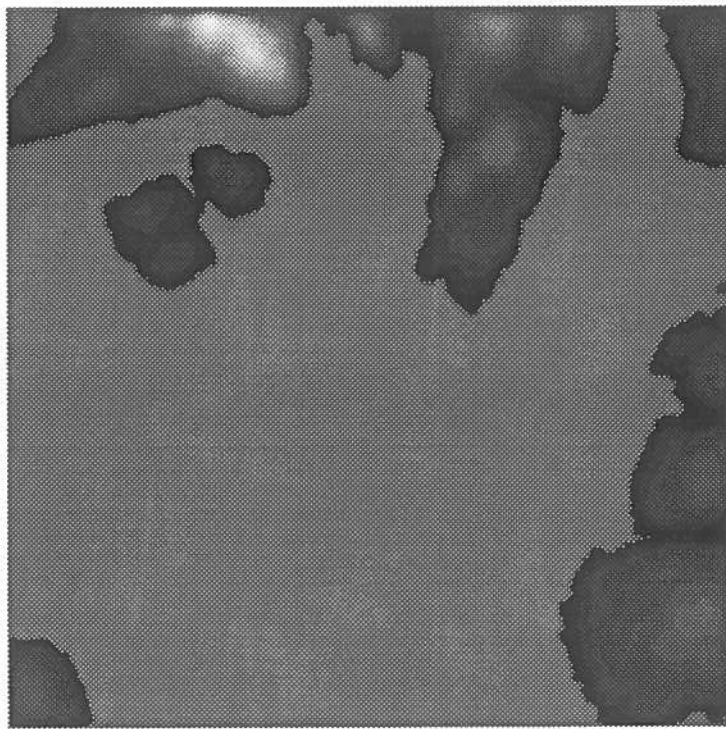
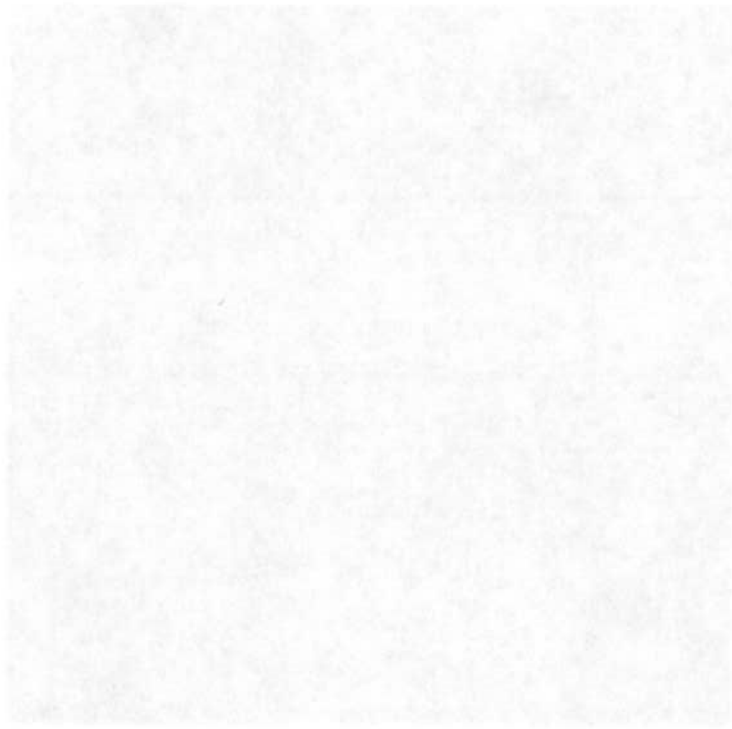


Figure B.7: NG40 Range Image (Smoothed $\times 40$).

The first part of the document discusses the importance of maintaining accurate records of all transactions. It emphasizes that proper record-keeping is essential for the success of any business and for the protection of the interests of all parties involved. The text outlines the various methods and systems used to collect, store, and analyze financial data, highlighting the need for consistency and transparency in the reporting process.

The second part of the document focuses on the role of technology in modern accounting. It explores how digital tools and software have revolutionized the way financial information is processed and communicated. The author discusses the benefits of automation, such as increased efficiency and reduced risk of human error, while also addressing the challenges of data security and privacy in a digital environment.

The final section of the document provides a comprehensive overview of the current state of the accounting profession. It examines the impact of global economic trends and regulatory changes on the industry, and offers insights into the future of accounting as a profession. The author concludes by emphasizing the importance of continuous learning and professional development for accountants in a rapidly changing world.



This figure illustrates the relationship between the variables discussed in the text, showing a clear trend that supports the author's conclusions.



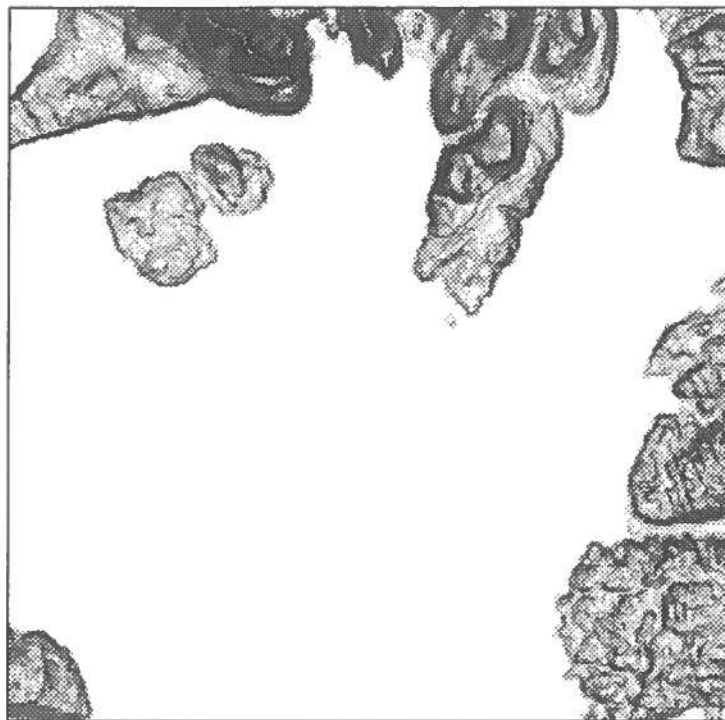


Figure B.8: NG40 *Cosine Shaded Image.*

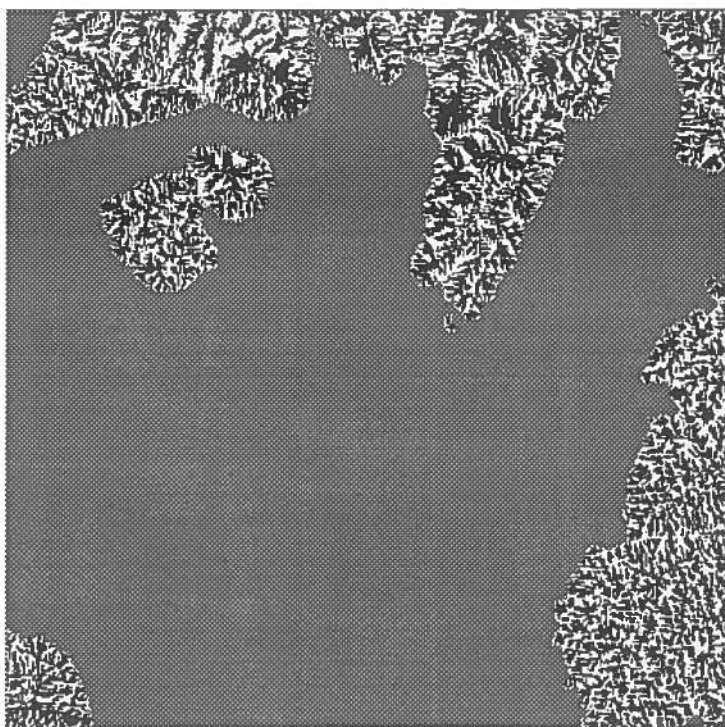
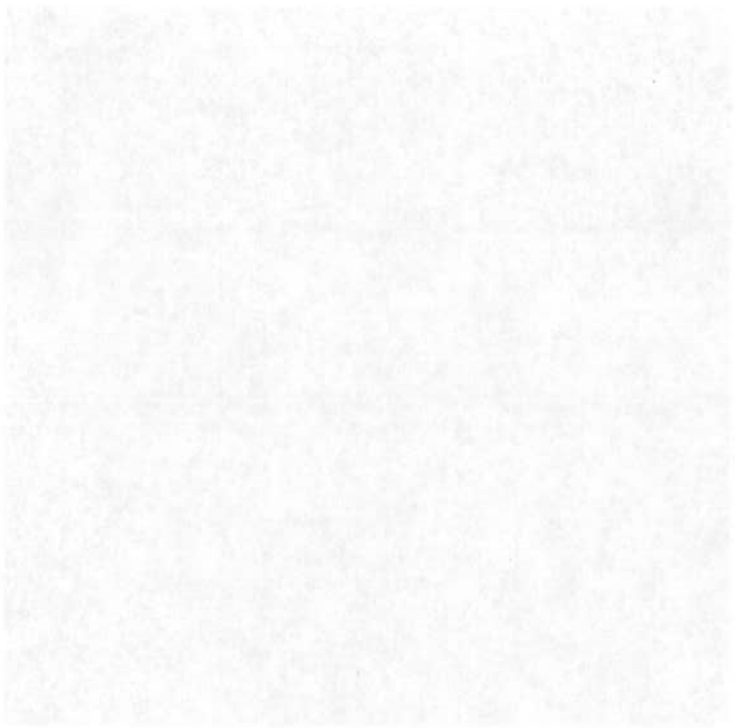
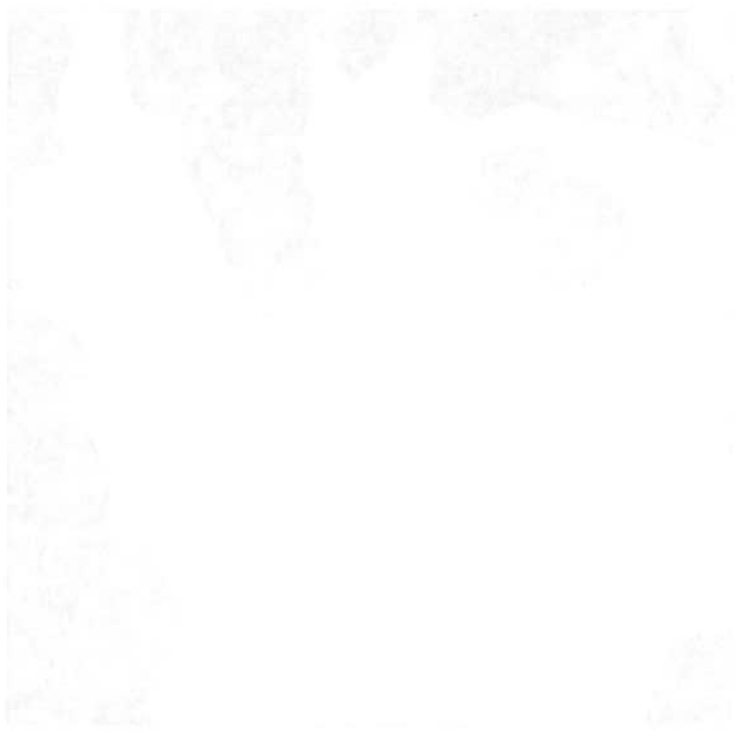


Figure B.9: NG40 *Classified Image (Original).*
white \Rightarrow *valley*; *black* \Rightarrow *ridge*; *grey* \Rightarrow *flat.*



Illegible text, possibly a title or header.



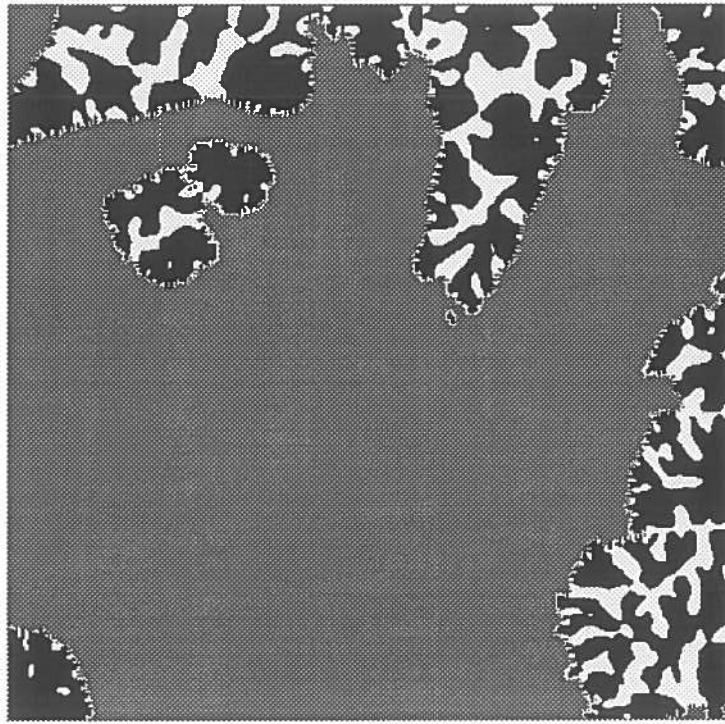


Figure B.10: NG40 *Classified Image (Smoothed $\times 40$)*.

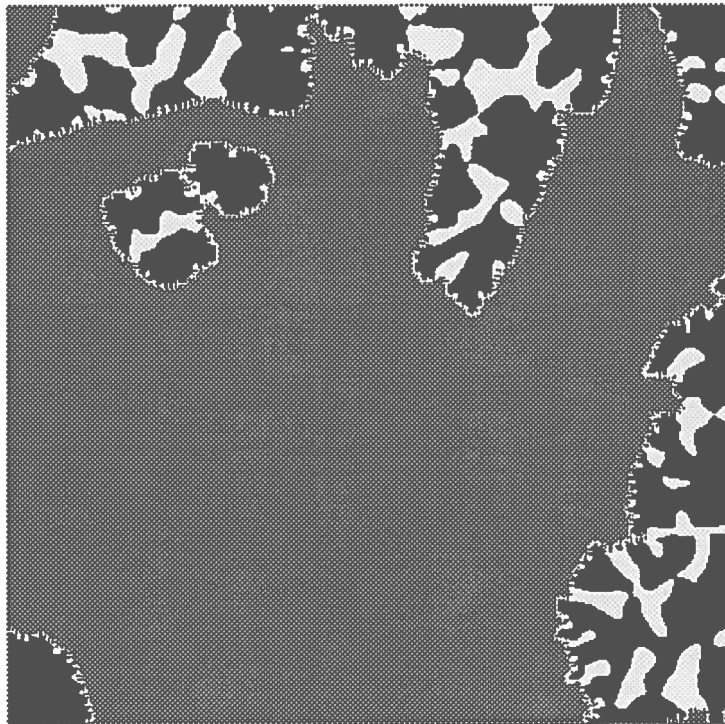
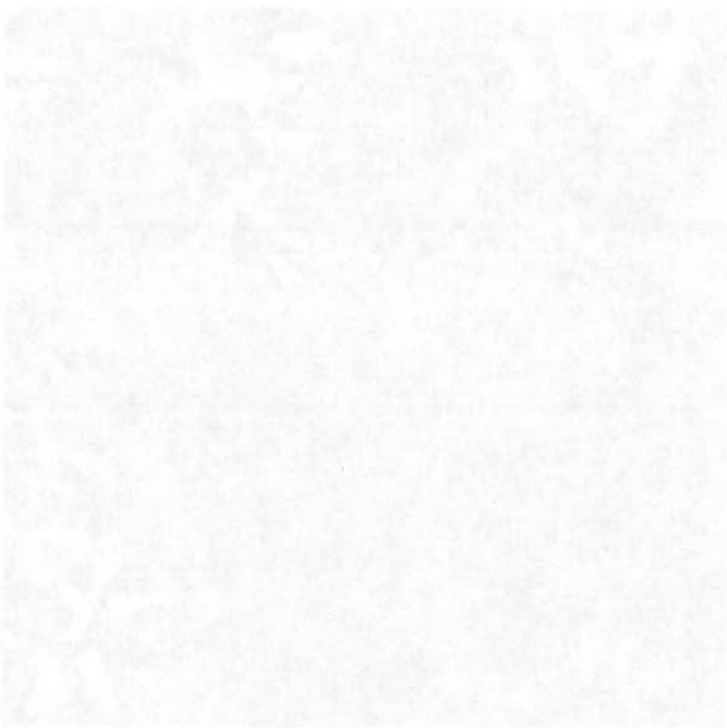
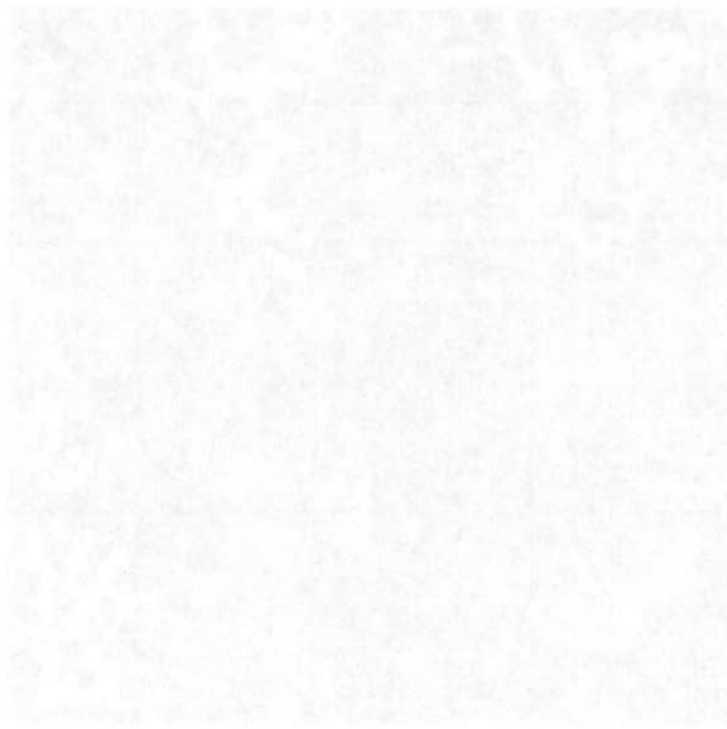


Figure B.11: NG40 *Classified Image (Smoothed $\times 80$)*.
white \Rightarrow valley; black \Rightarrow ridge; grey \Rightarrow flat.



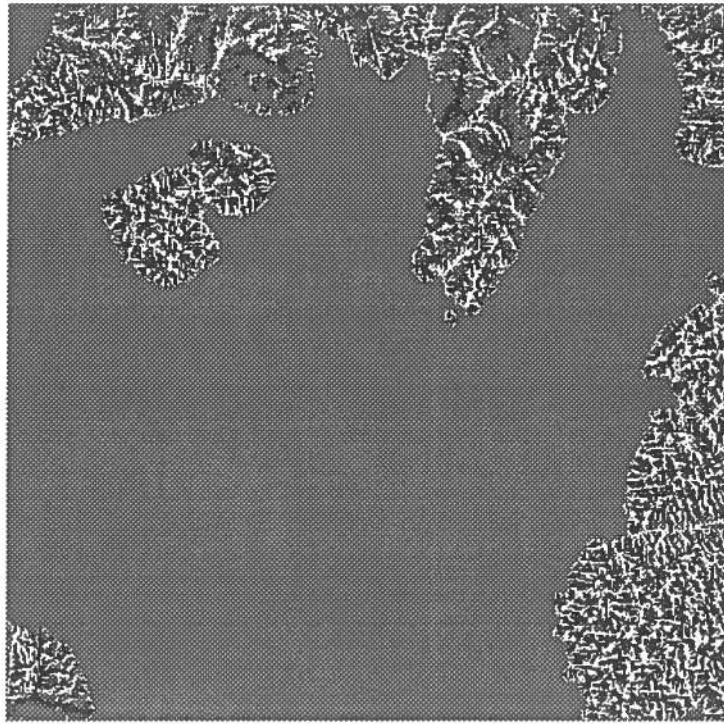


Figure B.12: *Thresholded NG40 Classified Image (Original).*

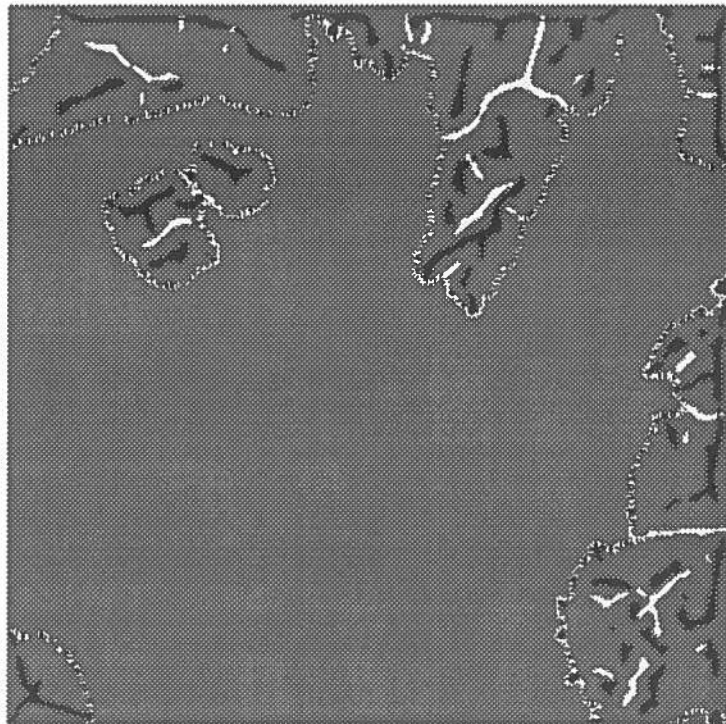


Figure B.13: *Thresholded NG40 Classified Image (Smoothed $\times 40$).*
white \Rightarrow valley; black \Rightarrow ridge; grey \Rightarrow flat.

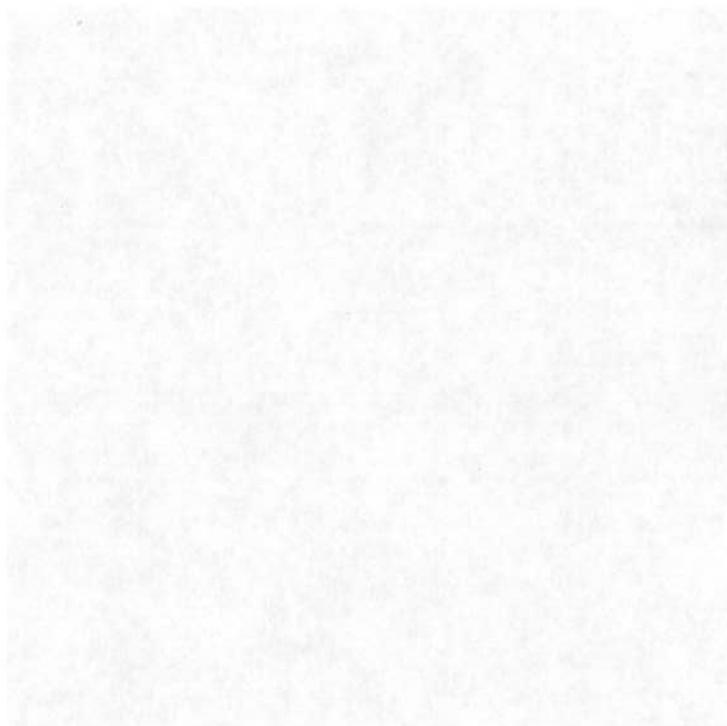
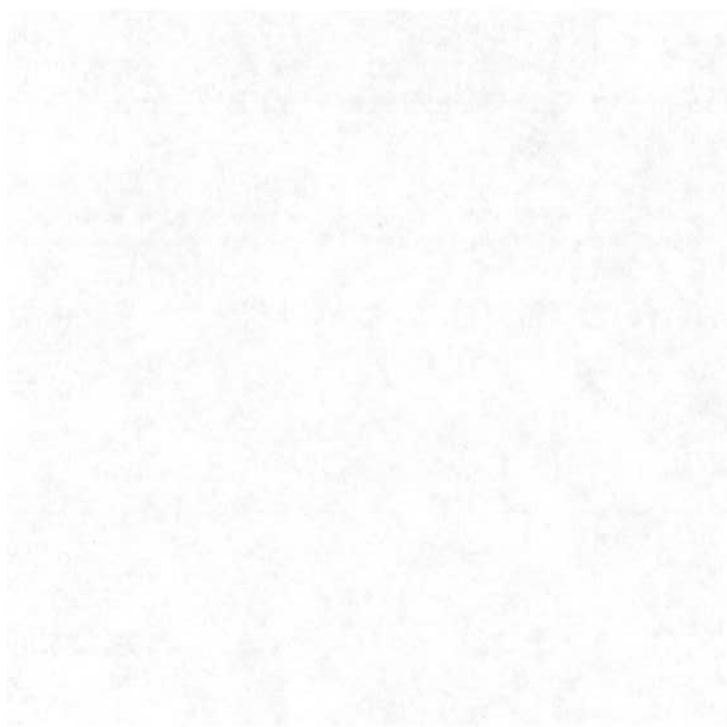


Figure 1. Comparison of the two images. The top image is the original image, and the bottom image is the image after processing.

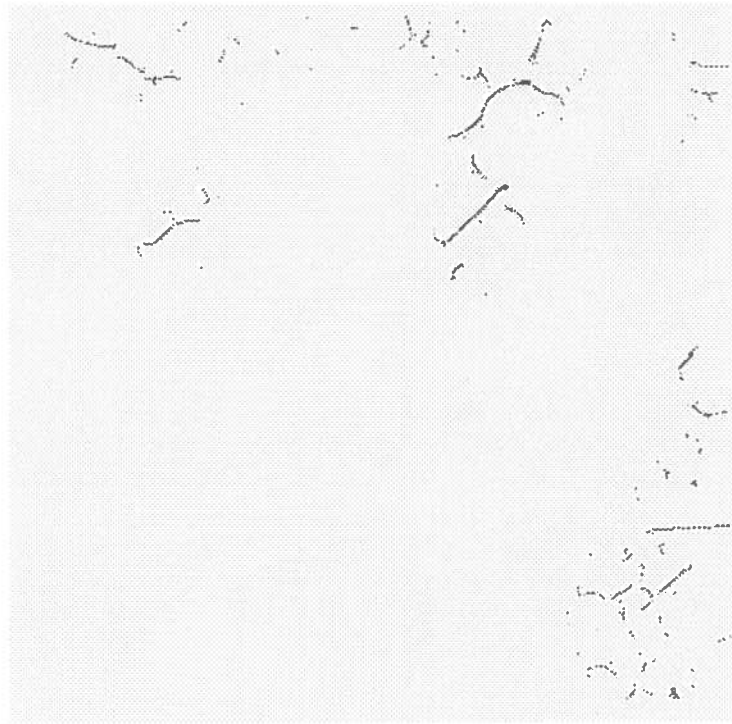


Figure B.14: NG40 (*Smoothed* $\times 40$) *Unconnected Valley Minima.*

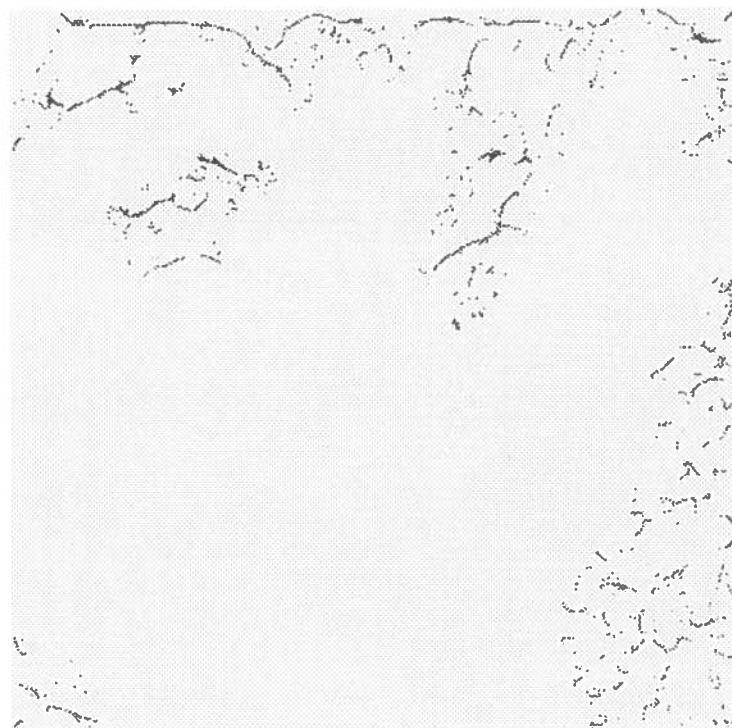


Figure B.15: NG40 (*Smoothed* $\times 40$) *Unconnected Ridge Apices.*

1. The first part of the document discusses the importance of maintaining accurate records of all transactions and activities. It emphasizes that this is crucial for ensuring transparency and accountability in the organization's operations.

2. The second part of the document outlines the various methods and tools used to collect and analyze data. It highlights the need for consistent data collection practices and the use of advanced analytical techniques to derive meaningful insights from the data.

3. The third part of the document focuses on the implementation of data-driven decision-making processes. It provides a detailed overview of the steps involved in identifying key performance indicators (KPIs) and using data to inform strategic decisions.

4. The fourth part of the document addresses the challenges and risks associated with data management. It discusses the importance of data security, privacy, and compliance with relevant regulations, and offers strategies to mitigate these risks.

5. The fifth part of the document concludes by summarizing the key findings and recommendations. It emphasizes the need for a continuous and iterative process of data analysis and decision-making to ensure the organization's long-term success.

© 2024, All rights reserved. | Confidential | Internal Use Only

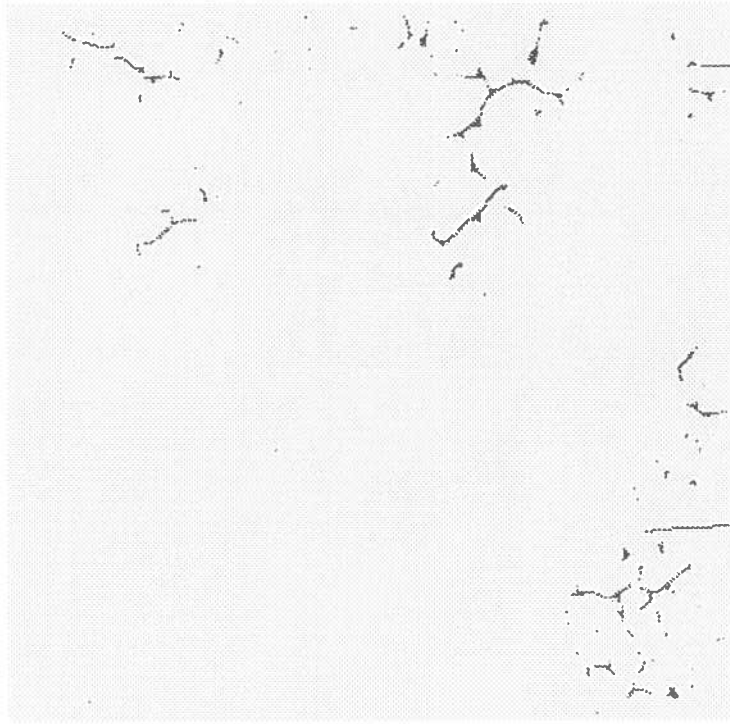


Figure B.16: *Connecting Gaps in the NG40 Valley Minima Tracks.*

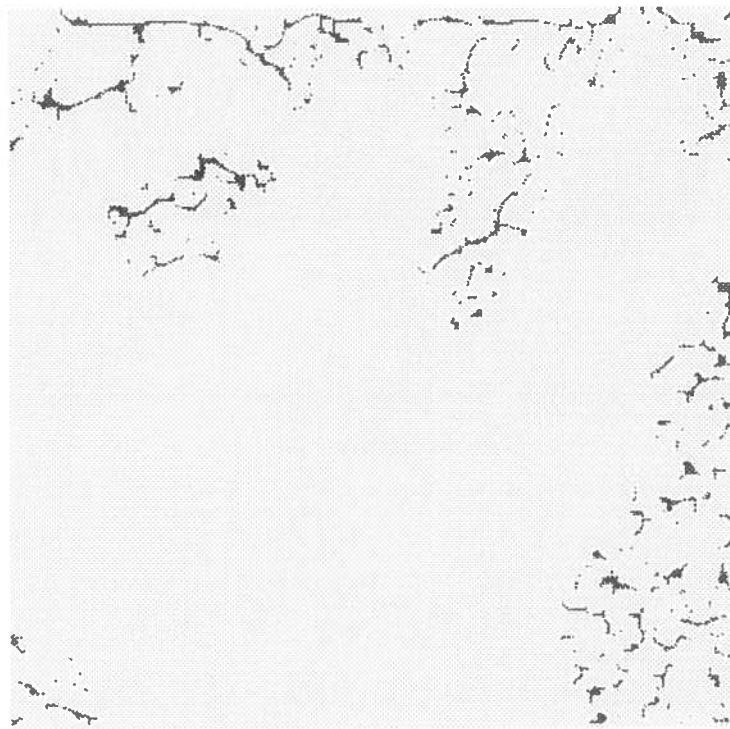


Figure B.17: *Connecting Gaps in the NG40 Ridge Apex Tracks.*

1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900

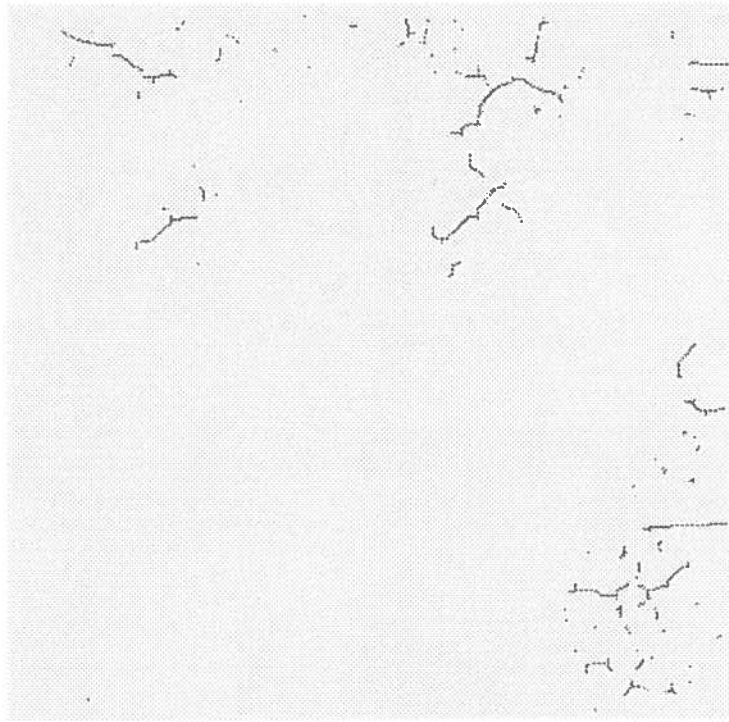


Figure B.18: *The Effect of Track Thinning (NG40 Valley Minima).*

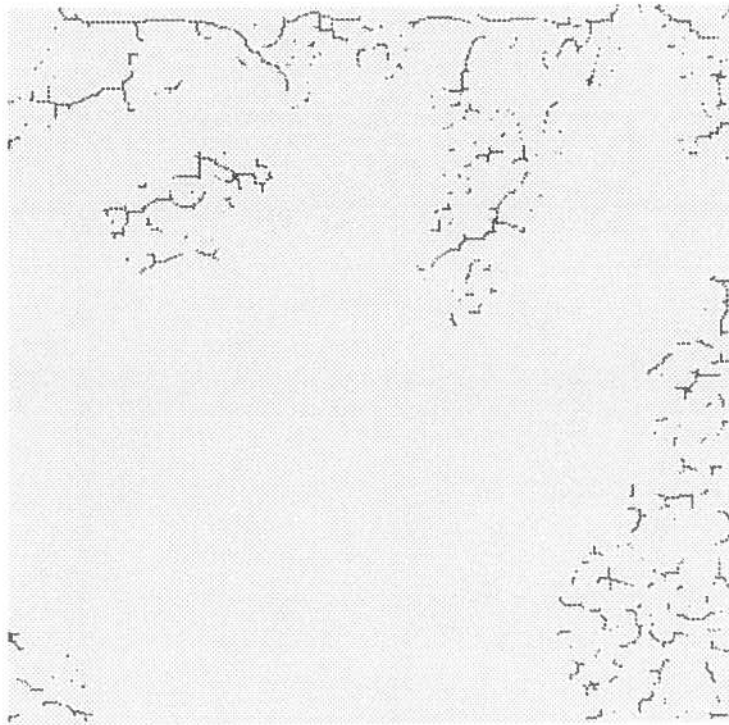


Figure B.19: *The Effect of Track Thinning (NG40 Ridge Apexes).*

THE UNIVERSITY OF CHICAGO



THE UNIVERSITY OF CHICAGO

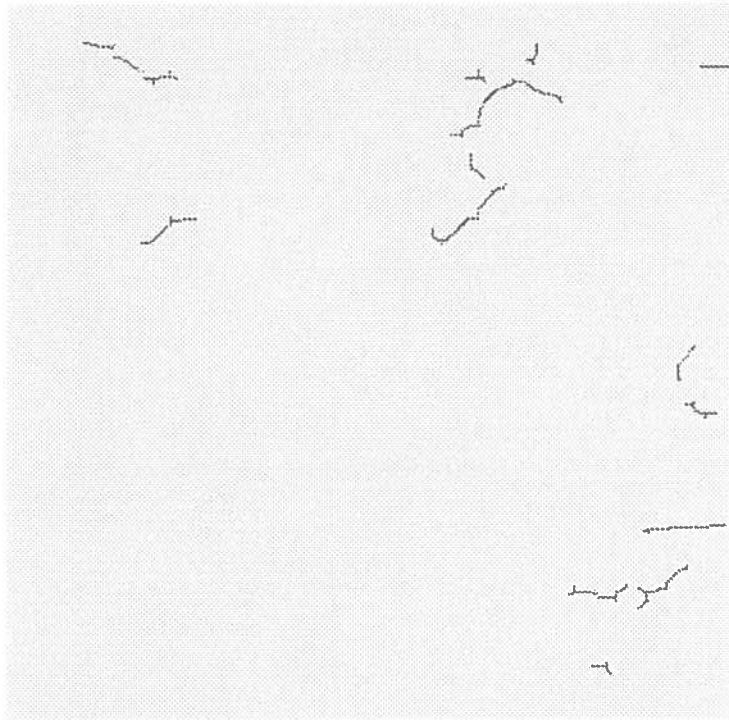


Figure B.20: NG40 *Valley Minima of < 20 Pixels Removed.*

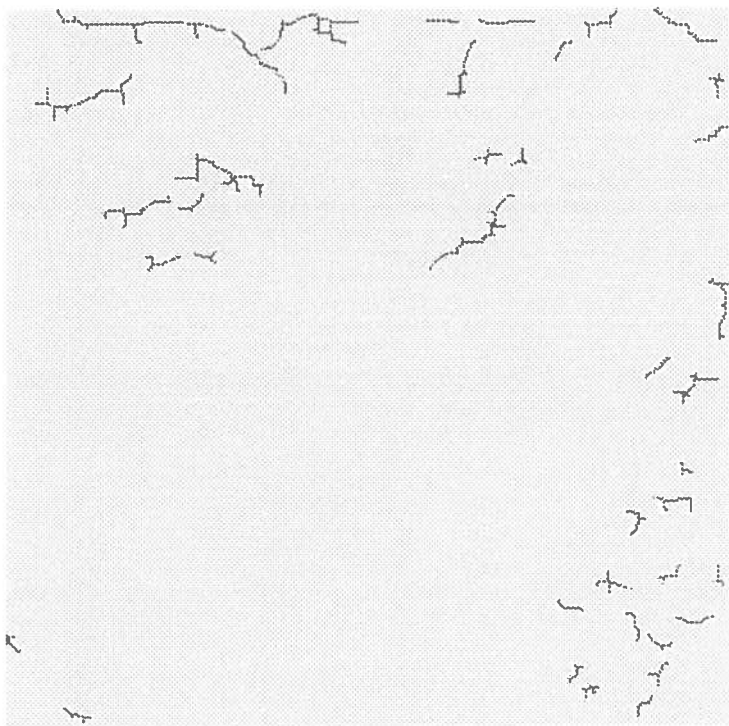


Figure B.21: NG40 *Ridge Apexes of < 20 Pixels Removed.*



1. The first part of the document is a
 2. description of the project and its
 3. objectives. It is followed by a
 4. detailed description of the work
 5. done during the period covered by
 6. the report. The next section is
 7. a summary of the results obtained
 8. and a discussion of their significance.
 9. The final section is a list of
 10. references.

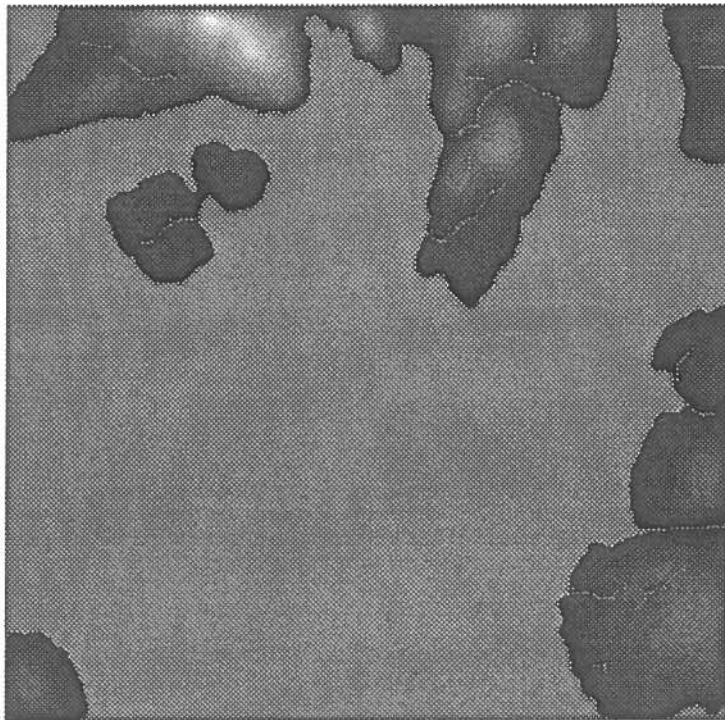


Figure B.22: NG40 *Overlaid Valley Minima.*

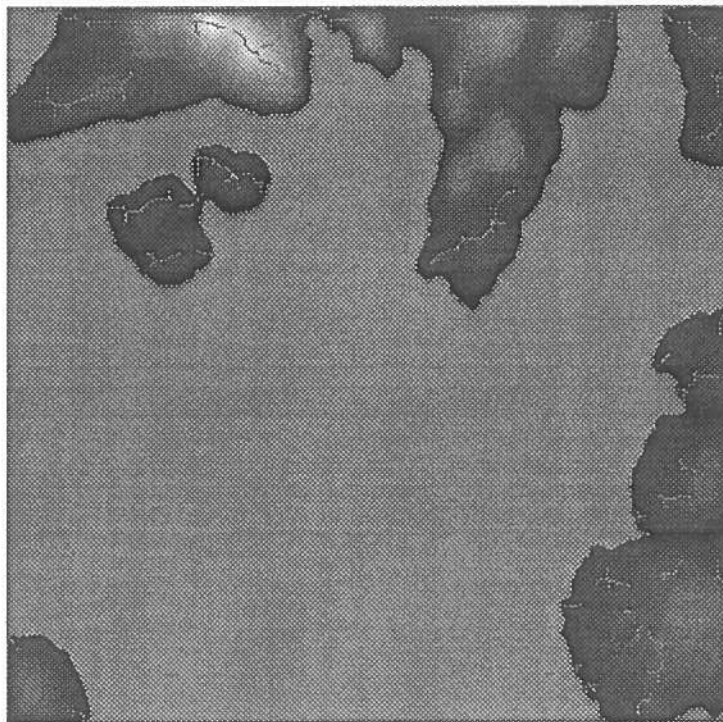
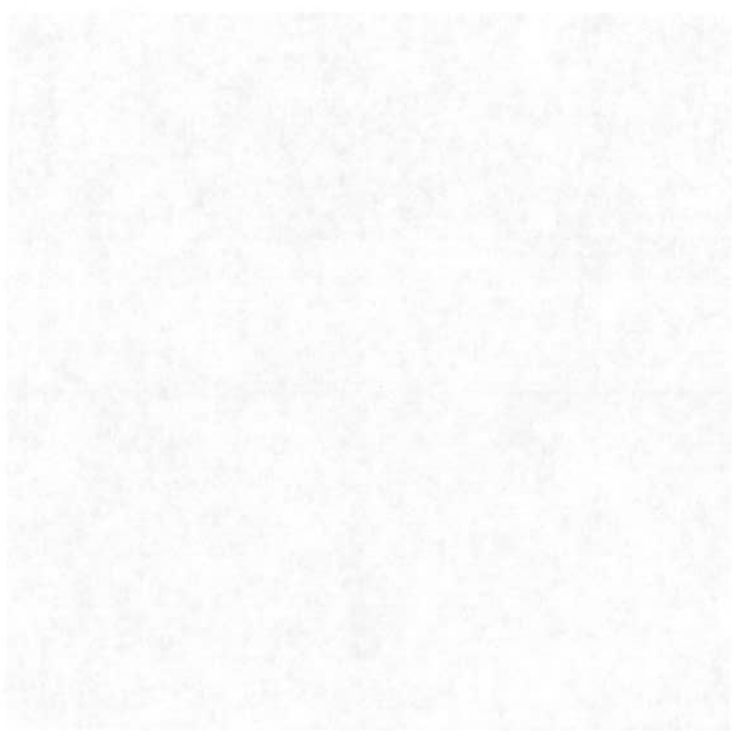
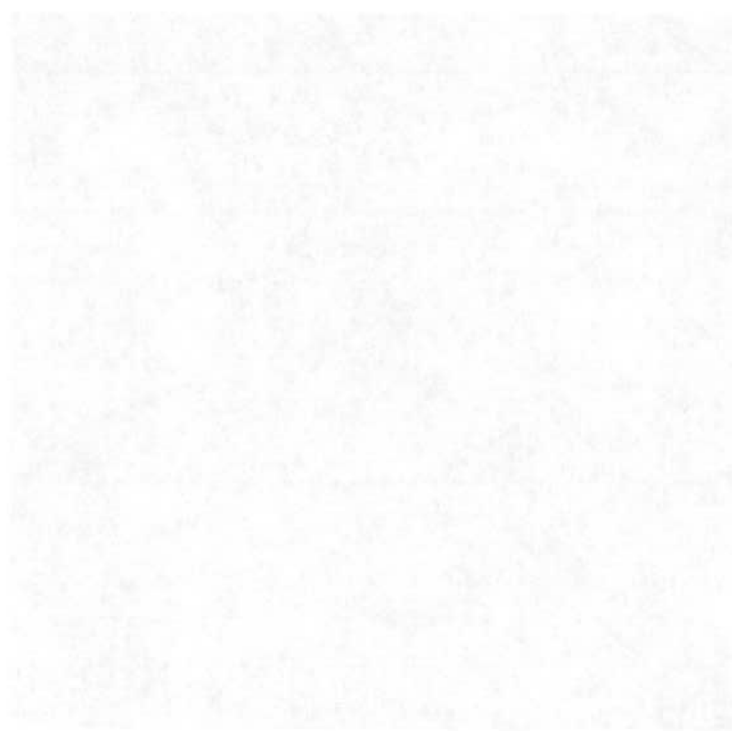


Figure B.23: NG40 *Overlaid Ridge Apexes.*



Appendix C

Program Code

C.1 Data Smoothing & Preprocessing

C.1.1 preprocess.cxx

1917

1918

1919

1920

1921

1922

1923

1924

1925

1926

1927

1928

1929

1930

1931

1932

```

/*
 * Preprocesses an NTF data file into a suitable
 * 400x400 pixel array of values, which is
 * smoothed via <int> iterations, as specified
 * by the -s command line value.
 *
 * usage : preprocess -i <infile> -o <outfile> -s int
 */
#include <stdio.h>

extern array2hips(float *, int, char [20]);

/*
 * This funtion reads 4 ASCII characters from
 * the file pointed to by fileptr, and
 * returns the corresponding integer value.
 */
float ascii_height(FILE *fileptr)
{
    char *cptr = (char *) malloc (4 * sizeof(char));
    cptr[0] =getc(fileptr);
    cptr[1] =getc(fileptr);
    cptr[2] =getc(fileptr);
    cptr[3] =getc(fileptr);
    return ((float) atoi(cptr));
}

/*
 * Function to smooth the original data using
 * a 3x3 convolution operator mask. The mask
 * is fitted around each pixel, with the
 * centre pixel getting the sum of the neigh-
 * bouring pixel products divided by 24.
 */
1 2 1
2 12 2
1 2 1

The smoothing is done a number of times
specified by 'iterations', which was given
as a command line argument.

Data coords range from 0:400 and 0:400
although we loop from 1:399 and 1:399
as the mask cannot be applied at the
extreme edges of the image. The height
values are in the range -100:1500m.
*/
void smooth(float **data, int iterations)
{
    int i, j, loop;
    float **newdata;

    /* allocate memory for updated values array */
    newdata = (float **) malloc (401 * sizeof (float *));
    newdata[0] = (float *) malloc (401 * 401 * sizeof (float));
    for (i = 1; i < 401; i++)
        newdata[i] = newdata[0] + i * 401;

    for (loop = 0; loop < iterations; loop++)
    {
        fprintf(stderr, "preprocess : smoothing iteration %d / %d\n",

```

```

        loop+1, iterations);
    }
    for (i = 1; i < 400; i++)
    {
        for (j = 1; j < 400; j++)
        {
            if (data[i][j] == 0) /* ignore sea-level pixels */
            {
                newdata[i][j] = 0;
                continue;
            }
            newdata[i][j] = data[i-1][j-1] +
                data[i+1][j-1] +
                data[i-1][j+1] +
                data[i+1][j+1] +
                data[i][j-1] * 2 +
                data[i][j+1] * 2 +
                data[i-1][j] * 2 +
                data[i+1][j] * 2 +
                data[i][j] * 12;

            newdata[i][j] /= 24;
        }
    }
    for (i = 0; i < 401; i++)
        for (j = 0; j < 401; j++) /* next iteration */
            data[i][j] = newdata[i][j]; /* use new values */
}

/*
 * Main program begin.
 */
main(int argc, char *argv[])
{
    int i, j, count, loop, iterations;
    FILE *in_fileptr, *out_fileptr;
    char *rubbish = (char *) malloc (80 * sizeof (char)),
        out_file[20];
    float **data;

    if (argc != 7)
    {
        fprintf(stderr, "usage : data2hips -i <file> -o <file> -s int\n");
        exit(0);
    }

    for (loop = 1; loop < argc; loop++) /* check command line args */
    {
        if (argv[loop][0] == '-')
        {
            switch (argv[loop][1])
            {
                case 'o' :
                    sscanf(argv[loop+1], "%s", out_file);
                    break;
                case 'i' :
                    sscanf(argv[loop+1], "%s", in_file);
                    break;
                case 's' :
                    sscanf(argv[loop+1], "%d", &iterations);

```

The first part of the paper is devoted to a review of the literature on the topic. It is found that there is a general consensus that the current system of international law is inadequate to deal with the challenges of the 21st century. The authors argue that a new system of international law is needed, one that is based on the principles of justice, equity, and fairness.

The authors propose a new system of international law that is based on the principles of justice, equity, and fairness. This new system would be based on the following principles:

1. The principle of justice: This principle requires that all states be treated equally under the law.
2. The principle of equity: This principle requires that the law be applied in a way that is fair and equitable to all states.
3. The principle of fairness: This principle requires that the law be applied in a way that is fair and equitable to all individuals.

The authors argue that this new system of international law is necessary in order to deal with the challenges of the 21st century. They believe that the current system of international law is inadequate to deal with the challenges of the 21st century, and that a new system of international law is needed.

The authors conclude that the current system of international law is inadequate to deal with the challenges of the 21st century, and that a new system of international law is needed. They believe that the new system of international law that they propose is the best way to deal with the challenges of the 21st century.

The second part of the paper is devoted to a discussion of the implications of the authors' proposals. They argue that the new system of international law would have a number of important implications for the world.

First, the new system of international law would be more just, equitable, and fair than the current system. This would be a significant improvement over the current system, which is often criticized for being unjust, inequitable, and unfair.

Second, the new system of international law would be more effective than the current system. This is because the new system would be based on the principles of justice, equity, and fairness, which are more likely to be accepted and followed by all states and individuals.

Third, the new system of international law would be more stable than the current system. This is because the new system would be based on the principles of justice, equity, and fairness, which are more likely to be accepted and followed by all states and individuals, leading to a more stable and peaceful world.

The authors conclude that the new system of international law that they propose is the best way to deal with the challenges of the 21st century. They believe that the new system of international law is necessary in order to deal with the challenges of the 21st century, and that the new system of international law that they propose is the best way to deal with the challenges of the 21st century.

```
break;
default : /* error arg */
  printf(stderr, "usage : data2hips -i <file> -o <file> -s int\
n");
  }
  }
  }
  if ((in_fileptr = fopen(in_file, "r")) == NULL)
  {
    fprintf(stderr, "data2hips : can't open input file : %s\n", in_file);
    exit(0);
  }
  /* allocate memory dynamically */
  data = (float **) malloc (401 * sizeof (float *));
  data[0] = (float *) malloc (401 * 401 * sizeof (float));
  for (i = 1; i < 401; i++)
    data[i] = data[0] + i * 401;

  for (loop = 0; loop < 401; loop++) /* # of data blocks */
  {
    fprintf(stderr, "preprocess : reading block[%d]\n", loop);

    count = 400;
    fgets(rubbish, 80, in_fileptr); /* ignore '51 xxx' line */
    for (j = 0; j < 21; j++) /* 21 lines per block */
    {
      for (i = 0; i < 19; i++) /* 19 values per line */
        data[count--][loop] = ascii_height(in_fileptr);

      fgets(rubbish, 10, in_fileptr); /* ignore ending 'l' char */
    }

    for (i = 1; i >= 0; i--) /* final two values */
      data[i][loop] = ascii_height(in_fileptr);

    fgets(rubbish, 10, in_fileptr); /* ignore ending '0' char */
  }

  fclose(in_fileptr);

  if (iterations > 0) smooth(data, iterations);

  printf("preprocess : creating HIPS image\n");
  array2hips("data", 401, out_file); /* create HIPS image */
}
```



Very faint text at the bottom right of the page, possibly a page number or footer, which is illegible due to fading.



C.2 Local Curvature, Shape & Orientation Calculation

C.2.1 process.cxx

C.2.2 hkcode.cxx

C.2.3 orient.cxx


```

/*
 * Process a HIPS image taken from stdin, by doing
 * the following steps:
 *
 * [1] : Uses a winsize * winsize window to look at
 *       each image pixel.
 * [2] : Fits a biquadratic surface about each pixel
 *       (if possible), using data from the window.
 * [3] : Calculates the mean curvature (H) value for
 *       each pixel using the biquadratic fit.
 * [4] : Calculates the tan of the angle that the
 *       minimum curvature of each pixel makes
 *       for use during suppression.
 * [5] : Dumps the array of H and tan values to
 *       separate files for use during suppression.
 * [6] : Generates, and dumps to file, a cosine shaded
 *       image.
 *
 * Usage : process -i file_id [-w integer] < HIPS_image
 *
 * The -i flag specifies the identifier which will
 * be used to tag all dump files and HIPS images
 * that are generated along the way.
 *
 * The -w option specifies the n x n size of the
 * window which is used to fit the biquadratic
 * surface. The default size is 3 (the window is
 * always square).
 *
 */
#include <stdio.h>
#include <string.h>
#include <hipl_format.h>
#define FALSE 0
#define TRUE 1

char Progname[]="process"; // needed by <hipl_format.h>

extern float calc_H(double *par, float x, float y);
extern double tancalc(double *par, float x, float y, float *coshaded);
extern array2hips(float *data, int side, char file[20], float min, float max);

// Andrew Fitzgibbon's library SVD Stuff

extern double **nr_matrix(int nrl,int nrh,int ncl,int nch);
extern double *nr_vector(int nl,int nh);
extern void nr_free_matrix(double **m,int nrl,int nrh,int ncl,int /* nch*/);
extern void nr_free_vector(double *v,int nl,int /*nh*/);
extern void build_svd_stuff(int offset, int npts, double ** U, double ** v, double *
w);
extern void svbksb(double ** u, double * w, double ** v, int m, int n, double * b, do
uble * x);

/*
 * Function to apply window from top to
 * bottom, left to right. Generates a
 * list of height values at each of the
 * [X,Y] points within the window.
 *
 * The function is passed the 'answer' into
 * which to place the required results. This
 * is allocated the required memory at
 * the beginning of the main program, and
 * freed accordingly at the end of processing.

```

```

*/
void apply_window(int left, int top, int right, int bottom, int cols,
float *data, double *Zlist)
{
    int step = 1;
    for (int i = left; i <= right; i++)
        for (int j = top; j <= bottom; j++)
            Zlist[step++] = data[i * cols + j];
}

/*
 * Function to check that the window
 * does not access the array of image
 * pixels out of bounds, ie. values
 * less than zero or > max which would
 * cause an error. If part of window
 * falls out of range then we cannot
 * calculate the H or K value for
 * that window's pixel.
 */
int window_in_range(int a, int b, int x, int y, int max)
{
    if ( ( a < 0) || ( b < 0) || ( x > max) || ( y > max) )
        return FALSE; // out of range
    return TRUE; // in range
}

/*
 * Function to dump the arrays of H and tan
 * values to separate files, named by the
 * following convention, according to the
 * user give file_id (specified through the
 * -i command line argument) :
 *
 * h_identifier (for the H array)
 * t_identifier (for the tan array)
 *
 */
void dump_to_file(int size, float *H, float *T, char file_id[20])
{
    FILE *H_fileptr, *T_fileptr;
    char h[22] = "h";
    char t[22] = "t";
    strcat(h, file_id); // generate the filenames
    strcat(t, file_id);
    if ((H_fileptr = fopen(h, "w")) == NULL) // open files if poss
    {
        if ((T_fileptr = fopen(t, "w")) == NULL)
        {
            fprintf(stderr, "process : can't open dump files\n");
            exit(0);
        }
    }
    fprintf(stderr, "\n\n process : dumping details to files\n");
    fprintf(H_fileptr, "%d\n", size); // first dump array sizes
    fprintf(T_fileptr, "%d\n", size);
    for (int loop = 0; loop < size; loop++) // then all the values
    {
        fprintf(H_fileptr, "%g\n", H[loop]);
        fprintf(T_fileptr, "%g\n", T[loop]);
    }
}

```

The first part of the paper discusses the importance of maintaining accurate records of all transactions. This is particularly true for businesses that operate in a highly competitive market. By keeping detailed records, a business can identify areas where costs are being incurred unnecessarily and take steps to reduce them.

In addition, accurate records are essential for determining the true profitability of a business. Without proper record-keeping, it is difficult to know whether a business is actually making a profit or operating at a loss. This information is crucial for making informed decisions about the future of the business.

The second part of the paper focuses on the importance of maintaining accurate records of all transactions. This is particularly true for businesses that operate in a highly competitive market. By keeping detailed records, a business can identify areas where costs are being incurred unnecessarily and take steps to reduce them.

In addition, accurate records are essential for determining the true profitability of a business. Without proper record-keeping, it is difficult to know whether a business is actually making a profit or operating at a loss. This information is crucial for making informed decisions about the future of the business.

The third part of the paper discusses the importance of maintaining accurate records of all transactions. This is particularly true for businesses that operate in a highly competitive market. By keeping detailed records, a business can identify areas where costs are being incurred unnecessarily and take steps to reduce them.

In addition, accurate records are essential for determining the true profitability of a business. Without proper record-keeping, it is difficult to know whether a business is actually making a profit or operating at a loss. This information is crucial for making informed decisions about the future of the business.

The fourth part of the paper focuses on the importance of maintaining accurate records of all transactions. This is particularly true for businesses that operate in a highly competitive market. By keeping detailed records, a business can identify areas where costs are being incurred unnecessarily and take steps to reduce them.

In addition, accurate records are essential for determining the true profitability of a business. Without proper record-keeping, it is difficult to know whether a business is actually making a profit or operating at a loss. This information is crucial for making informed decisions about the future of the business.

```

fclose(H_fileptr);
fclose(T_fileptr);
}

/*
 *
 * main program begin
 */
main(int argc, char *argv[])
{
    struct header hd;
    char file_id[20];
    float shade;
    int curr_row = 500, loop,
        picsize,
        winsize = 3,
        x, y,
        offset,
        xbegin, xend,
        ybegin, yend,
        no_of_points;

    if (argc < 3)
    {
        fprintf(stderr, "usage : process -i file_id [-w integer] < HIPS_image
        \n");
        exit(0);
    }
    for (loop = 1; loop < argc; loop++)
    {
        if (argv[loop][0] == '-')
        {
            switch (argv[loop][1])
            {
                case 'w' :
                    sscanf(argv[++loop], "%d", &winsize);
                    break;
                case 'i' :
                    sscanf(argv[++loop], "%s", file_id);
                    break;
                default :
                    fprintf(stderr, "usage : process -i file_id [-w integer] < HI
                    PS_image\n");
                    exit(0);
                    break;
            }
        }
        read_header(&hd);
        if (hd.pixel_format != PFFLOAT)
        {
            fprintf(stderr, "process : incorrect image pixel format\n");
            exit(0);
        }
        offset = (winsize - 1) / 2;
        picsize = hd.rows * hd.cols;
        no_of_points = winsize * winsize;
        fprintf(stderr, "\n Window Size : %d\n", no_of_points);
        fprintf(stderr, " Picture Size : %d\n", picsize);
        double *biquad_params = new double [6]; // memory allocation

```

```

float * H = new float [picsize];
float * T = new float [picsize];
float * S = new float [picsize];
float * data = new float [picsize];

pread(0, data, picsize*sizeof(float)); // read in HIPS data

/*
 *
 * allocate space for matrices and
 * vectors to be used in the SVD
 * fit (Andrew FG's code) and then
 * generate matrix U and vectors V, W
 * needed for the biquadratic fit.
 */
double ** U = nr_matrix(1, no_of_points, 1, 6);
double ** V = nr_matrix(1, 6, 1, 6);
double ** W = nr_vector(1, 6);
double ** Z = nr_vector(1, no_of_points);
build_svd_stuff(offset, no_of_points, U, V, W);

// process every image pixel
for (loop = 0; loop < picsize; loop++)
{
    y = loop % hd.cols; // calc 2D coords from 1D data
    x = (loop - y) / hd.cols;
    if (curr_row != x) { // per row progress report
        curr_row = x;
        if ((loop % 10 == 0)
            fprintf(stderr, "\n Processing Row : %3d", curr_row);
            else fprintf(stderr, " %3d", curr_row);
        }
    // calculate window limits and check
    xbegin = x - offset; xend = x + offset;
    ybegin = y - offset; yend = y + offset;
    if (!window_in_range(xbegin, ybegin, xend, yend, hd.cols - 1))
    {
        H[loop] = T[loop] = -9; // dummy values
        continue; // next pixel in loop
    }
    else
    {
        // apply the window to get the list of height (z) values
        apply_window(xbegin, ybegin, xend, yend, hd.cols, data, Z);
        svbksb(U, W, V, no_of_points, 6, Z, biquad_params-1);
        // calculate mean curvature (H) value
        // the tan of the major axis
        // and the cosine shaded pixel value
        H[loop] = calc_H(biquad_params, x, y);
        T[loop] = (float) tan(calc(biquad_params, x, y, &shade));
        S[loop] = shade;
    }
}
// generate and dump the cosine shaded HIPS image
// with the filename in accord with the identifier :
// identifier_shaded (the image filename)
char result[30] = "";
```

1. The first part of the document discusses the importance of maintaining accurate records of all transactions. It emphasizes that every entry should be supported by a valid receipt or invoice.

2. The second part details the various methods used to collect and analyze data. This includes both manual and automated processes, ensuring that the information gathered is reliable and up-to-date.

3. The third section focuses on the implementation of new software systems. It outlines the steps from selection to deployment, highlighting the need for thorough testing and user training.

4. The final part of the document addresses the challenges of data security and privacy. It provides recommendations for protecting sensitive information and complying with relevant regulations.

5. In conclusion, the document stresses the importance of a proactive approach to data management. By following the guidelines outlined here, organizations can ensure the integrity and security of their information.

Page 1 of 1
 Date: 10/26/2023
 Author: J. Smith

The following table provides a summary of the key findings from the recent audit. It shows a significant increase in revenue over the last quarter, which is a positive indicator for the company's growth.

However, there were also some areas of concern, particularly regarding the accuracy of the financial reporting. It is recommended that the accounting department review the current procedures to ensure that all transactions are recorded correctly.

Additionally, the audit identified several opportunities for cost reduction. By streamlining certain processes and negotiating better terms with suppliers, the company can improve its overall financial performance.

Overall, the audit has provided valuable insights into the company's financial health. It is essential that these findings are acted upon promptly to address any issues and capitalize on the identified opportunities.

Page 2 of 2
 Date: 10/26/2023
 Author: J. Smith

```
strcpy(result, file_id);
strcat(result, "_shaded");
array2hips(s, hd_cols, result, 0, 1);
dump_to_file(picsize, H, T, file_id); // dump the arrays to file
// free allocated memory
delete H;
delete T;
delete S;
delete data;
delete biquad_params;
nr_free_matrix(U, 1, no_of_points, 1, 6);
nr_free_matrix(V, 1, 6, 1, 6);
nr_free_vector(W, 1, 6);
nr_free_vector(Z, 1, no_of_points);
fprintf(stderr, "\n");
}
```

<p style="text-align: center;"> 1. Name of the person or organization 2. Address 3. City 4. State 5. Zip 6. Telephone 7. Fax 8. E-mail 9. Other </p>	10. Date 11. Page
--	----------------------


```

/*
 * This file contains functions to calculate the mean (H)
 * and gaussian (K) curvatures at some given point on a
 * biquadratic fit. The arguments required are :
 *
 * (1) an array of the biquadratic parameters
 * (2) x coord (usually zero due to transposing)
 * (3) y coord (usually zero due to transposing)
 *
 * To calculate the two principle curvatures, use:
 *
 * P1 = H - sqrt(H*H-K)
 * P2 = H + sqrt(H*H-K)
 *
 */
#include <math.h>
float calc_K(double *par, float x, float y);
float calc_H(double *par, float x, float y);
float calc_K(double *par, float x, float y)
{
    float a,b,c,d,e,f;          /* the various derivatives of z */
    float zxx,zyy,zxy,zx,zy;   /* Gaussian curvature */
    float K;
    a=par[0];
    b=par[1];
    c=par[2];
    d=par[3];
    e=par[4];
    f=par[5];
    zxx=2*d;
    zyy=2*f;
    zxy=f;
    zx=b;
    zy=c;
    K = ( zxx*zyy-zxy*zxy ) / ( (1+zx*zx+zy*zy)*(1+zx*zx+zy*zy) );
    return K;
}
float calc_H(double *par, float x, float y)
{
    float a,b,c,d,e,f;
    float zxx,zyy,zxy,zx,zy;   /* the various derivatives of z */
    float H;
    float top,bottom;
    a=par[0];
    b=par[1];
    c=par[2];
    d=par[3];
    e=par[4];
    f=par[5];
    zxx=2*d;
    zyy=2*f;
    zxy=f;
    zx=b;
    zy=c;
    // b + 2*d*x + f*y      (but x = 0 = y)
    // c + 2*e*y + f*x      (but x = 0 = y)
}

```

```

top = zxx+zyy+zxx*zxy+zyy*zx-zx-2*zx*zy*zxy;
bottom = 2*(sqrt(1+zx*zx+zy*zy))*sqrt(1+zx*zx+zy*zy)*(sqrt(1+zx*zx+zy*zy));
H = top/bottom;
return H;
}

```



```

/*
 * This function calculates the tan of the
 * angle that the major curvature axis
 * makes with the x-axis, (towards the
 * y-axis).
 *
 * This is done by working out both the
 * principal curvature magnitudes and
 * choosing the smaller in order to work
 * out the (u,v)-direction vectors of
 * the minimum curvature.
 *
 * The tan the minimum curvature axis makes
 * is then just :
 *
 *      tan B = v / u
 *
 * We require three arguments :
 *
 * [1] : the array of biquadratic parameters.
 * [2] : the x-coordinate.
 * [3] : the y-coordinate.
 *
 * The function also calculates the cosine
 * shaded pixel value in order to build up
 * an overall cosine shaded image, as it
 * requires much of the same maths needed
 * in calculating the curvatures etc.
 */
#include<math.h>
double tancalc(double *par, float x, float y, float *coshaded)
{
    double  a, b, c, d, e, f,
           root, top,
           zx, zy, zxx, zyy, zxy;

    a = par[0]; b = par[1]; c = par[2];
    d = par[3]; e = par[4]; f = par[5];

    zx = b;
    zy = c;
    zxx = 2*d;
    zxy = f;
    zyy = 2*e;

    double E = 1 + zx * zx;
    double F = zx * zy;
    double G = 1 + zy * zy;

    // generate the cosine shaded pixel value
    root = sqrt(1 + zx*zx + zy*zy);
    *coshaded = (float) 1 / root;

    double L = zxx / root;
    double M = zxy / root;
    double N = zyy / root;

    double S = E*G - F*F;
    double A = G*L - F*M;
    double B = G*M - F*N;
    double C = E*M - F*L;
    double D = E*N - F*M;

```

```

    root = sqrt( (D - A)*(D - A) + 4*B*C );
    double C1 = (D + A + root) / 2;
    double C2 = (D + A - root) / 2;
    if (fabs(S*C1 - A) < 1e-10)
        return 0;
    if (fabs(B) < 1e-10)
        return infinity();
    // Compare magnitudes and return the tan
    // the smallest one makes with the x-axis.
    // This is the minimum curvature, ie. the
    // curvature along the feature
    if (fabs(C1) < fabs(C2))
        return ((S*C1 - A) / B);
    else
        return ((S*C2 - A) / B);
}

```

The first part of the document discusses the importance of maintaining accurate records of all transactions. It emphasizes that every entry should be supported by a valid receipt or invoice. This not only helps in tracking expenses but also ensures compliance with tax regulations.

In the second section, the author outlines the various methods used for data collection and analysis. These include surveys, interviews, and focus groups. Each method has its own strengths and weaknesses, and the choice depends on the specific needs of the study.

The third section provides a detailed overview of the results obtained from the research. It highlights key findings and trends, as well as any unexpected observations. The data suggests that there is a significant correlation between the variables being studied, which supports the initial hypothesis.

Finally, the document concludes with a summary of the findings and offers some practical recommendations for future research. It suggests that further exploration is needed in certain areas to gain a more comprehensive understanding of the subject matter.

The following table summarizes the key data points from the study. It shows the distribution of responses across different categories, along with the corresponding percentages.

Category	Percentage
Category A	35%
Category B	25%
Category C	15%
Category D	10%
Category E	15%

The data indicates that Category A is the most prevalent, followed by Category B. Categories C, D, and E represent smaller but still significant portions of the total sample.

The author also notes that there were some limitations to the study, such as a relatively small sample size and potential biases in the data collection process. However, the findings are still considered valuable and provide a solid foundation for further research.

C.3 Non-Maximal Suppression & Tracking

C.3.1 `suppress.cxx`

C.3.2 `remove.cxx`

Journal of Management Studies

Volume 45, Number 1, February 2012

ISSN 0022-2525



```

/*
 * Performs a non-maximal suppression type
 * algorithm to identify the candidate points
 * which are the bottom of valleys and the
 * tops of ridges.
 *
 * Each pixel can be classified according to
 * the sign of its mean curvature (H) value
 * calculated during the image processing
 * stage, according to the following :
 *
 * H < 0 => ridge-type point
 * H > 0 => valley-type pixel
 * H = 0 => plane, minimal etc.
 *
 * These bands can be augmented using a
 * threshold value, to allow for almost
 * zero values, and/or to enable us to
 * ignore "weak" candidate pixels, ie :
 *
 * H < +threshold => ridge-type point
 * H > -threshold => valley-type pixel
 *
 * All in all, I suppose its tracking by
 * cheating - without using hysteresis,
 * gradient descent, or anything of the like !
 *
 * Usage : suppress -i file_id [-t threshold] < HIPS_image
 *
 * The -i flag specifies the identifier which will
 * be used to tag the two generated HIPS images of
 * the "tracked" valleys and ridges.
 *
 * The -t option allows the user to specify a
 * threshold value to be used in tolerating
 * +ve and -ve deviations from zero for the
 * mean curvature (H) values. There is a
 * built-in default of 0.0 (surprisingly !)
 *
 * */
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#include <hipl_format.h>

#define deg2rad(x) ((x * M_PI) / 180) // a nice conversion function
#define rad2deg(x) ((x * 180) / M_PI) // another nice function

#define TRUE 1
#define FALSE 0

char Progname[]="suppress"; // needed by <hipl_format.h>

extern array2hips(float *, int, char [20], float min, float max);

/*
 * Function returns TRUE iff value is strictly
 * less than or equal to both alpha and beta.
 */
float minimum(float value, float alpha, float beta)
{
    if ((value <= alpha) && (value <= beta))
        return TRUE;
    else

```

```

    return FALSE;
}

/*
 * Function returns TRUE iff value is strictly
 * greater than or equal to both alpha and beta.
 */
float maximum(float value, float alpha, float beta)
{
    if ((value >= alpha) && (value >= beta))
        return TRUE;
    else
        return FALSE;
}

/*
 * main program begin
 */
main(int argc, char *argv[])
{
    struct header hd;
    char file_id[20], thresh_id[8] = "0.0";
    FILE *t_fileptr, *h_fileptr;
    int t_size, h_size;
    double threshold = 0.0; // default value

    if (argc < 3)
    {
        fprintf(stderr, "usage : suppress -i file_id [-t threshold] < HIPS_im
age\n");
        exit(0);
    }
    for (int loop = 1; loop < argc; loop++) // check command line args
    {
        if (argv[loop][0] == '-')
        {
            switch (argv[loop][1])
            {
                case 't' : // specify threshold
                    sscanf(argv[loop+1], "%s", thresh_id);
                    threshold = atof(thresh_id);
                    fprintf(stderr, "\n h_threshold = %f\n", threshold);
                    break;
                case 'i' : // specify file identifier
                    sscanf(argv[loop+1], "%s", file_id);
                    break;
                default : // sorry, illegal arg
                    fprintf(stderr, "usage : suppress -i file_id [-t threshold] <
HIPS_image\n");
                    exit(0);
                    break;
            }
        }
    }
    char t_filename[22] = "t-";
    char h_filename[22] = "h-";
    strcat(t_filename, file_id);
    strcat(h_filename, file_id);
    if (((t_fileptr = fopen(t_filename, "r")) == NULL)
        || ((h_fileptr = fopen(h_filename, "r")) == NULL))

```

1. The first part of the document discusses the importance of maintaining accurate records of all transactions. This is essential for ensuring the integrity of the financial statements and for providing a clear audit trail.

2. The second part of the document outlines the various methods used to collect and analyze data. These methods include direct observation, interviews, and the use of statistical techniques. Each method has its own strengths and limitations, and it is important to choose the most appropriate one for the specific situation.

3. The third part of the document describes the process of data analysis. This involves identifying patterns, trends, and anomalies in the data. It also involves testing hypotheses and drawing conclusions based on the results.

4. The fourth part of the document discusses the importance of communication in the research process. This includes presenting the findings in a clear and concise manner, as well as responding to questions and feedback from others.

5. The fifth part of the document concludes with a summary of the key points discussed. It emphasizes the need for a systematic and rigorous approach to research, and the importance of transparency and honesty in reporting the results.

6. The sixth part of the document discusses the ethical considerations of research. This includes issues such as informed consent, confidentiality, and the potential for harm to participants. It is important to be aware of these issues and to take steps to minimize any potential risks.

7. The seventh part of the document discusses the practical aspects of conducting research. This includes issues such as budgeting, scheduling, and the selection of a research team. It is important to plan carefully and to be flexible in the face of unexpected challenges.

8. The eighth part of the document discusses the importance of staying up-to-date on the latest research in the field. This involves reading relevant journals, attending conferences, and participating in professional organizations.

9. The ninth part of the document discusses the importance of collaboration and teamwork in research. This involves working closely with colleagues, sharing ideas and resources, and supporting each other throughout the process.

10. The tenth part of the document concludes with a final summary of the key points discussed. It emphasizes the need for a holistic and integrated approach to research, and the importance of continuous learning and improvement.


```

{
    fprintf(stderr, "suppress : can't open dump files\n");
    exit(0);
}

fscanf(t_fileptr, "%d\n", &t_size); // read in array sizes
fscanf(h_fileptr, "%d\n", &h_size); // which must be the same
if (t_size != h_size)
{
    fprintf(stderr, "suppress : H & T file sizes incompatible\n");
}
else
{
    exit(0);
}

// Then lettuce begin !!
read_header(&hd);
if ((hd.pixel_format != PFFLOAT) || (t_size != (hd.cols * hd.rows)))
{
    fprintf(stderr, "suppress : incorrect HIPS image input - size ?\n");
    exit(0);
}

float h_value, ang_tan;

float *H = new float [t_size]; // memory allocation
float *T = new float [t_size];
float *data = new float [t_size];
float *val_trk = new float [t_size];
float *rdg_trk = new float [t_size];

pread(0, data, t_size*sizeof(float)); // read in HIPS data

int x, y, curr_row = (hd.cols + 2);
for (loop = 0; loop < h_size; loop++) // check every pixel
{
    y = loop & hd.cols; // calc 2D coords from 1D data
    x = (loop - y) / hd.cols;

    if (curr_row != x) { // per row progress report
        curr_row = x;
        if (loop % 10 == 0)
            fprintf(stderr, "\n Suppressing Row : %3d", curr_row);
        else
            fprintf(stderr, " %3d", curr_row);
    }

    fscanf(h_fileptr, "%g\n", &H[loop]); // get H value
    fscanf(t_fileptr, "%g\n", &T[loop]); // get tan value

    h_value = H[loop];
    ang_tan = T[loop];

    // first decide if pixel is a ridge or valley
    if ( ((h_value < threshold) && (h_value > (0 - threshold)))
        || (h_value == -9) // ignore "border" values
        || (data[loop] == 0) // ignore sea_level pixels
    )
    {
        rdg_trk[loop] = 0; // non-candidate ridge point
        val_trk[loop] = 0; // non-candidate valley point
        continue; // next pixel in loop
    }

    // actually the angle of orientation is anticlockwise
    // from the y-axis, so by adding 90 degrees, we get

```

```

// the angle clockwise from the x-axis, towards the
// y-axis which determines which case holds.

double angle = rad2deg(atan(ang_tan));
angle += 90;

if (angle < 0) angle += 180; // same differences
if (angle > 180) angle -= 180;

double alpha, beta, tan_angle;

/*
 * For non-maximal suppression,
 * there are four cases to be
 * considered :
 * [1] : 0 <= Angles <= 45
 * [2] : 45 < Angles <= 90
 * [3] : 90 < Angles <= 135
 * [4] : 135 < Angles <= 180
 *
 * At each stage we calculate
 * two interpolated height values,
 * alpha and beta, which lie
 * along the line of curvature
 * represented by the tan of the
 * angle that it makes with the
 * x-axis.
 *
 * For valley pixels (ie. H +ve)
 * the current pixel [x,y] must
 * be less than both alpha and
 * beta to survive suppression.
 *
 * For ridge pixels (ie. H -ve)
 * the current pixel [x,y] must
 * be greater than both alpha and
 * beta to survive suppression.
 *
 * The functions maximum and minimum
 * return either 0 and 1 corresponding
 * to a suppressed or non-suppressed
 * pixel.
 */

if (angle <= 45) // Case [1]
{
    tan_angle = tan(deg2rad(angle));
    alpha = (1 - tan_angle) * data[loop - hd.cols] +
            tan_angle * data[loop - hd.cols - 1];
    beta = (1 - tan_angle) * data[loop + hd.cols] +
            tan_angle * data[loop + hd.cols + 1];

    if (h_value > 0)
        val_trk[loop] = minimum(data[loop], alpha, beta);
    else
        if (h_value < 0)
            rdg_trk[loop] = maximum(data[loop], alpha, beta);
        else
            continue;
}

if (angle <= 90) // Case 2 : 90 degs
{

```

1. The first part of the document discusses the importance of maintaining accurate records of all transactions. This is essential for ensuring the integrity of the financial statements and for providing a clear audit trail.

2. The second part of the document outlines the various methods used to collect and analyze data. These methods include direct observation, interviews, and the use of specialized software tools.

3. The third part of the document describes the results of the data collection and analysis. It shows that there are significant areas where the current processes are inefficient and where improvements can be made.

4. The fourth part of the document provides recommendations for how to address these inefficiencies. These recommendations include implementing new software, reorganizing the workflow, and providing additional training for staff.

5. The fifth part of the document discusses the expected benefits of these improvements. These benefits include increased accuracy, reduced costs, and improved customer service.

6. The sixth part of the document concludes with a summary of the findings and a call to action for management to implement the recommended changes.

7. The seventh part of the document provides a detailed breakdown of the costs associated with the proposed improvements. This includes the cost of software licenses, hardware, and staff training.

8. The eighth part of the document discusses the timeline for implementing the improvements. It identifies key milestones and provides a realistic estimate of when the project will be completed.

9. The ninth part of the document provides a risk assessment for the proposed improvements. It identifies potential risks and provides strategies to mitigate these risks.

10. The tenth part of the document provides a final summary of the findings and recommendations. It emphasizes the importance of taking action on these findings to ensure the long-term success of the organization.

```

angle = 90 - angle;
tan_angle = tan(deg2rad(angle));

alpha = (1 - tan_angle) * data[loop - 1] +
tan_angle * data[loop - hd.cols - 1];

beta = (1 - tan_angle) * data[loop + 1] +
tan_angle * data[loop + hd.cols + 1];

if (h_value > 0)
    val_trk[loop] = minimum(data[loop], alpha, beta);
else
    if (h_value < 0)
        rdg_trk[loop] = maximum(data[loop], alpha, beta);
    continue;
}

if (angle <= 135) // Case 3 : 135 degs
{
    angle = angle - 90;
    tan_angle = tan(deg2rad(angle));

    alpha = (1 - tan_angle) * data[loop + 1] +
tan_angle * data[loop - hd.cols + 1];

    beta = (1 - tan_angle) * data[loop - 1] +
tan_angle * data[loop + hd.cols - 1];

    if (h_value > 0)
        val_trk[loop] = minimum(data[loop], alpha, beta);
    else
        if (h_value < 0)
            rdg_trk[loop] = maximum(data[loop], alpha, beta);
    continue;
}

if (angle <= 180) // Case 4 : 180 degs
{
    angle = 180 - angle;
    tan_angle = tan(deg2rad(angle));

    alpha = (1 - tan_angle) * data[loop - hd.cols] +
tan_angle * data[loop - hd.cols + 1];

    beta = (1 - tan_angle) * data[loop + hd.cols] +
tan_angle * data[loop + hd.cols - 1];

    if (h_value > 0)
        val_trk[loop] = minimum(data[loop], alpha, beta);
    else
        if (h_value < 0)
            rdg_trk[loop] = maximum(data[loop], alpha, beta);
    continue;
}

if (angle > 180) // A problem - shouldn't arise
{
    fprintf(stderr, "\n\nAngle > 180\n\n");
    continue;
}

}

fclose(h_fileptr); // close H and tan files as

```

```

fclose(h_fileptr);
fprintf(stderr, "\n\n");

/* // were finished with them

* * Generate tracked HIPS image filenames
* * and files. The filenames contain both
* * a specifier indicating whether the
* * HIPS image is of "tracked" valleys or
* * ridges and the threshold value used
* * (0.0) if none was indicated on the
* * command line, along with the usual
* * filename identifier used throughout, i.e. :
* * identifier_val_0.05
* * identifier_rdg_0.05
*/
char valleys[40] = "";
char ridges[40] = "";

strcpy(valleys, file_id);
strcat(valleys, "val");
strcat(valleys, thresh_id);

strcpy(ridges, file_id);
strcat(ridges, "rdg");
strcat(ridges, thresh_id);

array2hips(val_trk, hd.cols, valleys, 1, 0);
array2hips(rdg_trk, hd.cols, ridges, 1, 0);

// free allocated memory

delete H;
delete T;
delete data;
delete rdg_trk;
delete val_trk;
}

```

The first part of the paper discusses the importance of maintaining accurate records of all transactions. This is essential for the proper management of the company's finances and for ensuring compliance with tax laws. The second part of the paper discusses the importance of maintaining accurate records of all assets and liabilities. This is essential for the proper management of the company's balance sheet and for ensuring compliance with tax laws.

The third part of the paper discusses the importance of maintaining accurate records of all income and expenses. This is essential for the proper management of the company's profit and loss statement and for ensuring compliance with tax laws. The fourth part of the paper discusses the importance of maintaining accurate records of all tax payments. This is essential for the proper management of the company's tax liability and for ensuring compliance with tax laws.

```

/*
 * This code contains all the stack details
 * used to track all minima or maxima in an
 * image, removing isolated tracks with fewer
 * than a minimum number of pixels.
 *
 * usage : remove [-l int] < HIPS_track_image
 *
 * The -l option allows the user to indicate
 * a minimum length requirement. Tracks with
 * less than this amount of pixels in them
 * are deleted from the track image.
 *
 */
#include <stdio.h>
#include <stdlib.h>
#include <hipl_format.h>
extern array2hips(float *, int, char[20], float, float);
char Progname[]="remove"; // needed by <hipl_format.h>
struct stacknode // definition of a stack node
{
    int coord;
    struct stacknode *nextptr;
};
typedef struct stacknode SNODE;
typedef SNODE *SNODEPTR;

/*
 * push coordinate onto stack.
 */
void push(SNODEPTR *topptr, int coordinate)
{
    SNODEPTR newptr;

    newptr = malloc(sizeof(SNODE));
    if (newptr)
    {
        newptr->coord = coordinate;
        newptr->nextptr = *topptr;
        *topptr = newptr;
    }
    else
        fprintf(stderr, "No memory available for insertion.\n");
}

/*
 * pop coordinate (returned) from stack.
 */
int pop(SNODEPTR *topptr)
{
    SNODEPTR tempptr;
    int popcoord;

    tempptr = *topptr;
    popcoord = (*topptr)->coord;
    *topptr = (*topptr)->nextptr;
    free(tempptr);
    return popcoord;
}

```

```

}

/*
 * check emptiness of stack.
 */
int is_empty(SNODEPTR topptr)
{
    return !topptr;
}

/*
 * print stack contents
 */
void print_stack(SNODEPTR curptr)
{
    if (curptr == NULL)
        fprintf(stderr, "Stack Empty.\n");
    else
    {
        fprintf(stdout, ":\n");
        while (curptr != NULL)
        {
            fprintf(stdout, "%d ---> ", curptr->coord);
            curptr = curptr->nextptr;
        }

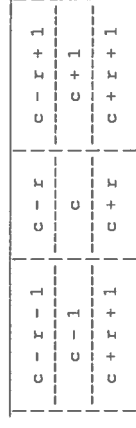
        fprintf(stdout, "NULL\n\n");
    }
}

/*
 * Free all memory held by a stack.
 */
void free_stack(SNODEPTR bogus_stack)
{
    int temp;

    while (!is_empty(bogus_stack))
    {
        temp = pop(&bogus_stack);
    }
}

/*
 * This function pushes the 8 neighbouring
 * coordinates of a point onto the stack.
 */
void push_8_neighbours(SNODEPTR *curptr, int coord, int row)
{
    push(curptr, coord - row - 1);
    push(curptr, coord - row);
    push(curptr, coord - row + 1);
}

```




```

push(currptr, coord - 1);
push(currptr, coord + 1);

push(currptr, coord + row - 1);
push(currptr, coord + row);
push(currptr, coord + row + 1);
}

/*
 * This is the function which takes the array of
 * pointers to the track stacks and produces the
 * required HIPS image.
 */
void generate_track_image(SNODEPTR *track, int size)
{
    int number = 0;

    float * data = new float [size*size];

    while (!is_empty(track[number]))
    {
        while (is_empty(track[number]))
            data[pop(track[number])] = 1;
        number++;
    }

    array2hips(data, size, "Removed", 1, 0);
}

/*
 * This function builds up an array of all the
 * tracks in an image which are at least min_length
 * long. Each array element is in fact a
 * pointer to a stack which contains all
 * the coordinates of the points along
 * that track:
 */
array { track 1 | track 2 | ..... | track n
-----|-----|-----|-----|-----
      coord 1
      coord 2
      coord n
      NULL
}

void remove_tracks(char *array, int size, int min_length)
{
    SNODEPTR * track = new SNODEPTR [500]; // array of track
    SNODEPTR stackptr = NULL;

    // push(stackptr, value);
    // pop(stackptr);
    // is_empty(stackptr);
    // print_stack(stackptr);

    int coord, number = 0, tracklength;

```

```

for (int c = 1; c < size - 1; c++)
for (int r = 1; r < size - 1; r++)
{
    tracklength = 0; // length of current track
    track[number] = stackptr; // specific track array element

    coord = r * size + c;
    if (array[coord] != 0) continue; // 0 => identified trackpt.

    array[coord] = -2; // point now considered
    tracklength++; // increase track length
    push(track[number], coord); // record coord on track stack
    push_8_neighbours(stackptr, coord, size); // get the 8 neighbours
    while (!is_empty(stackptr))
    {
        coord = pop(stackptr); // check next neighbour
        if (array[coord] != 0) continue; // 0 => a trackpt.
        array[coord] = -2; // remove
        tracklength++; // increase length
        push(track[number], coord); // record coord
        push_8_neighbours(stackptr, coord, size); // get 8 neighbours
    }

    if (tracklength < min_length)
        free_stack(track[number]); // free memory
    else
    {
        fprintf(stdout, "%d points on track ", tracklength);
        print_stack(track[number]);
        number++; // begin next track in sequence
    }

    fprintf(stdout, "Overall no. of track = %d\n", number);
    generate_track_image(track, size);
}

/*
 * Main program begin.
 * Accepts minimum length as command line
 * argument, otherwise it defaults to 0.
 */
main(int argc, char *argv[])
{
    int epsilon = 0; // default minimum length

    for (int loop = 1; loop < argc; loop++)
    {
        if (argv[loop][0] == '-')
        {
            switch (argv[loop][1])
            {
                case 'l' : // user supplied minimum length
                    sscanf(argv[loop+1], "%d", &epsilon);
                    break;
                default : // command line arg error
                    fprintf(stderr, "usage : remove [-l int] < HIPS_track
                    _file\n");
            }
        }
    }
}

```

1. The first part of the paper is a review of the literature on the topic of [faint text]. This section discusses the various methods used to study [faint text] and the results of these studies. It also discusses the limitations of these studies and the need for further research.

2. The second part of the paper is a description of the experimental design used in this study. This section describes the participants, the materials, and the procedures used in the study. It also describes the measures used to assess the dependent variables.

3. The third part of the paper is a presentation of the results of the study. This section presents the data from the study and discusses the findings. It also discusses the implications of these findings for the field of [faint text].

4. The fourth part of the paper is a discussion of the results and their implications. This section discusses the limitations of the study and the need for further research. It also discusses the implications of these findings for the field of [faint text].

5. The fifth part of the paper is a conclusion. This section summarizes the findings of the study and discusses the implications of these findings for the field of [faint text].

6. The sixth part of the paper is a list of references. This section lists the sources used in the study.

7. The seventh part of the paper is an appendix. This section contains additional information related to the study.

8. The eighth part of the paper is a list of figures. This section lists the figures included in the study.

9. The ninth part of the paper is a list of tables. This section lists the tables included in the study.

10. The tenth part of the paper is a list of footnotes. This section contains footnotes related to the study.

11. The eleventh part of the paper is a list of acknowledgments. This section acknowledges the individuals and organizations that supported the study.

12. The twelfth part of the paper is a list of disclosures. This section discloses any potential conflicts of interest.

13. The thirteenth part of the paper is a list of disclosures. This section discloses any potential conflicts of interest.

14. The fourteenth part of the paper is a list of disclosures. This section discloses any potential conflicts of interest.

15. The fifteenth part of the paper is a list of disclosures. This section discloses any potential conflicts of interest.

16. The sixteenth part of the paper is a list of disclosures. This section discloses any potential conflicts of interest.

17. The seventeenth part of the paper is a list of disclosures. This section discloses any potential conflicts of interest.

18. The eighteenth part of the paper is a list of disclosures. This section discloses any potential conflicts of interest.

19. The nineteenth part of the paper is a list of disclosures. This section discloses any potential conflicts of interest.

20. The twentieth part of the paper is a list of disclosures. This section discloses any potential conflicts of interest.


```
        }
    }
    exit(0);
    break;

    struct header hd;
    read_header(&hd);
    if (hd.pixel_format != PFBYTE)
    {
        fprintf(stderr, "remove: incorrect HIPS image input\n");
        exit(0);
    }
    int picsize = hd.cols * hd.rows;
    char * track_data = new char [picsize]; // allocate memory for data
    fread(0, track_data, picsize*sizeof(char)); // read in HIPS data
    fprintf(stdout, "Minimum Allowable Length = %d\n", epsilon);
    remove_tracks(track_data, hd.cols, epsilon); // remove small tracks
    delete track_data;
    // finished with it
}
```



C.4 Sundry

C.4.1 `hipl_format.h`

C.4.2 `array2hips.cxx`

1911

1912

1913



```

/* HIPL Picture Header Format Standard
 * Michael Landy - 2/1/82
 */

struct header {
    char *orig_name; /* The originator of this sequence */
    char *seq_name; /* The name of this sequence */
    int num_frames; /* The number of frames in this sequence */
    char *orig_date; /* The date the sequence was originated */
    int rows; /* The number of rows in each image */
    int cols; /* The number of columns in each image */
    int bits_per_pixel; /* The number of significant bits per pixel */
    int bit_packing; /* Nonzero if bits were packed contiguously */
    int pixel_format; /* The format of each pixel, see below */
    char *seq_history; /* The sequence's history of transformations */
    char *seq_desc; /* Descriptive information */
};

/* Pixel Format Codes
 */
#define PFBYTE 0 /* Bytes interpreted as integers */
#define PFSHORT 1 /* Short int's interpreted as integers */
#define PFINTE 2 /* Int's */
#define PFFLOAT 3 /* Float's */
#define PFCOMPLEX 4 /* 2 Float's interpreted as (real,imaginary) */
#define PFASCII 5 /* Ascii representation, with linefeeds after each ro
w */
#define PFOCT 11 /* quad-tree encoding */
#define PFBHIST 12 /* histogram of byte image */
#define PFSPAN 13 /* spanning tree format */
#define PFIOT3D 24 /* plot-3d format */
#define PFAHC 400 /* adaptive hierarchical encoding */
#define PFOCT 401 /* oct-tree encoding */
#define PFBT 402 /* binary tree encoding */
#define PFAHC3 403 /* 3-d adaptive hierarchical encoding */
#define PFBQ 404 /* binquad encoding */
#define PFRLED 500 /* run-length encoding */
#define PFRLEB 501 /* run-length encoding, line begins black */
#define PFRLEW 502 /* run-length encoding, line begins white */
/* the following were added for the AI dept */
#define PFTRE 503 /* tracked edge format */
#define PFCODE 504 /* line coding */

/* Bit packing formats
 */
#define MSBFIRST 1 /* bit packing - most significant bit first */
#define LSBFIRST 2 /* bit packing - least significant bit first */

#define FBUFFLIMIT 30000

/* For general readability
 */
#define TRUE 1
#define FALSE 0
typedef int Boolean;
/* extra bits tacked on in the AI dept */

```

```

/* text placement data */
struct text_print {int isize,ix,iy;} tp;

#ifdef _IBMPCL
#include <malloc.h>
#endif

```

Name: _____
 Section: _____
 Date: _____
 Title: _____
 Objectives: _____
 Introduction: _____
 Theory: _____
 Procedure: _____
 Results: _____
 Discussion: _____
 Conclusion: _____
 References: _____

Name: _____
 Section: _____
 Date: _____
 Title: _____
 Objectives: _____
 Introduction: _____
 Theory: _____
 Procedure: _____
 Results: _____
 Discussion: _____
 Conclusion: _____
 References: _____

```
/*
 * Routine to take a data array of float values
 * and scale the range from minimum : maximum
 * into the required HIPS byte range 0 : 255
 * and creates the correspondingly scaled HIPS
 * image in the specified file.
 */
#include <stdio.h>
#include <hipl_format.h>
// char Progname[] = "array2hips";

/*
 * Function which scales the values ranging
 * from minimum : maximum into the 0 : 255 range.
 */
int scale(float value, float minimum, float maximum)
{
    return (int) ((value - minimum) / ((maximum - minimum) / 255));
}

/*
 * Function which scales data into a bytesize
 * HIPS image (values ranging from 0 : 255)
 * factors depending on the maximum and minimum
 * data value.
 */
void array2hips(float *data, int size, char file[20], float min, float max)
{
    unsigned char *bytearray;
    int i, sfd, picsize;
    struct header hd;

    picsize = size * size;

    if ((sfd = creat(file, 0666)) < 0)
    {
        fprintf(stderr, "array2hips : can't open output file %s\n", file);
        exit(0);
    }

    init_header(&hd, file, "", "1", size, size, 8, 0, PFBYTE, "");
    write_header(sfd, &hd);

    if ((bytearray = (unsigned char *) calloc (picsize, 1))
        == (unsigned char *) NULL)
    {
        fprintf(stderr, "array2hips : can't allocate HIPS data space\n");
        exit(0);
    }

    for (i = 0; i < picsize; i++)
        bytearray[i] = scale(data[i], min, max);

    if (write(sfd, bytearray, picsize) != picsize)
    {
        fprintf(stderr, "array2hips : HIPS data write error\n");
        exit(0);
    }
}
```

The first part of the document discusses the importance of maintaining accurate records of all transactions. It emphasizes that every entry should be supported by a valid receipt or invoice. This not only helps in tracking expenses but also ensures compliance with tax regulations.

In the second section, the author provides a detailed breakdown of the monthly budget. It lists various categories such as housing, utilities, food, and transportation, along with their respective costs. This helps in understanding the overall financial picture and identifying areas where savings can be made.

The third part of the document focuses on investment strategies. It discusses the benefits of diversifying one's portfolio and the importance of long-term planning. The author suggests consulting with a financial advisor to tailor an investment plan that aligns with individual goals and risk tolerance.

Finally, the document concludes with a summary of key points and a call to action. It encourages readers to take control of their finances, review their records regularly, and make informed decisions about their future.