A COMPLICATED AND IMPRESSIVE SOUNDING TITLE THAT IS TOO LONG FOR A SINGLE LINE WHILE INCLUDING EVERYTHING

by

John Q. Engineer

A dissertation submitted in partial fulfillment
of the requirements for the degree

of

DOCTOR OF PHILOSOPHY

in

Electrical Engineering

Approved:

Richard P. Feynmann, Ph.D. Major Professor	Robert L. Forward, Ph.D. Committee Member
Albert Einstein, Ph.D. Committee Member	Gottfried Liebniz, Ph.D. Committee Member
Isaac Newton, Ph.D. Committee Member	D. Richard Cutler, Ph.D. Vice Provost of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2022

ABSTRACT

A Complicated and Impressive Sounding Title that is Too Long For a Single Line While Including Everything

by

John Q. Engineer, Doctor of Philosophy

Utah State University, 2022

Major Professor: Richard P. Feynmann, Ph.D.
Department: Electrical and Computer Engineering

This is the abstract of the demonstration thesis. Hopefully the examples will be sufficiently clear that you will have few formatting problems. The Graduate School requires that the abstract be 350 words or less, so be careful of the length of the abstract.

(34 pages)

PUBLIC ABSTRACT

A Complicated and Impressive Sounding Title that is Too Long For a Single Line While Including Everything

John Q. Engineer

The public abstract is to convey the purpose of the research to the PUBLIC, so layman's terms should be used.

To all the little people....

ACKNOWLEDGMENTS

I am so happy that my advisor helped me.....

John Q. Engineer

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACRONYMS

BFCS	body-fixed coordinate system
CEF	composite energy function (related to ILC)
CSOIS	Center for Self-Organizing and Intelligent Systems
CV	certainty value (related to HIMM)
DOF	degree of freedom
EKF	extended Kalman filter
FOG	fiber optic gyro
FOV	field of view of a camera
GAIC	geometric Akaike information criterion
GMDL	geometric minimum description length criterion
GRO	growth rate operator (related to HIMM)
HIMM	histogram in-motion mapping
HOSA	higher-order spectral analysis (related to Matlab toolbox)
IBO	identifier-based observer (related to PDS)
IIC	identical initial condition (related to ILC)
ILC	iterative learning control
ICS	inertial coordinate system
LAO	linear approximation-based observer (related to PDS)
LQG	linear quadratic Gaussian
LS	least squares
LTV	linear time-varying
NN	neural network
OCS	obstacle cluster strength (related to HIMM)
ODIS	omni-directional inspection system, a robot at the CSOIS center
ODV	omni-directional vehicle

CHAPTER 1

INTRODUCTION

Image compression or image coding is the process of reducing the redundancy in the image data that may result in some loss of information. Vector quantization (VQ)[1] is one such technique.

1.1 Background

Binary splitting is illustrated in Fig. 1.1.

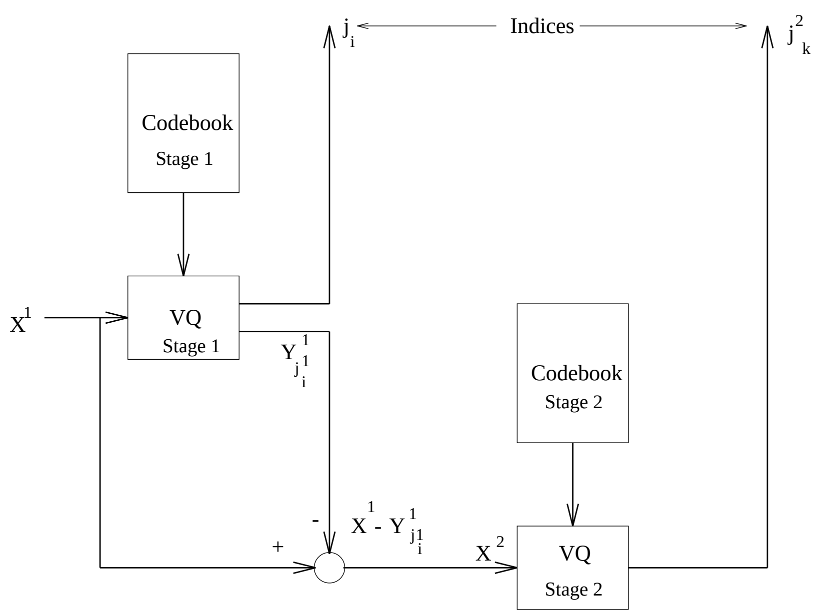

Fig. 1.1: Binary splitting.

This figure is generated using an open-source figure drawing package (called `fig`). Any figure drawing package can be used to generate figures. The easiest format for output is to output the figures in `.pdf` format for inclusion in the `.tex` file.

[1] The acronym VQ is used as an abbreviation for both vector quantization and vector quantizer.

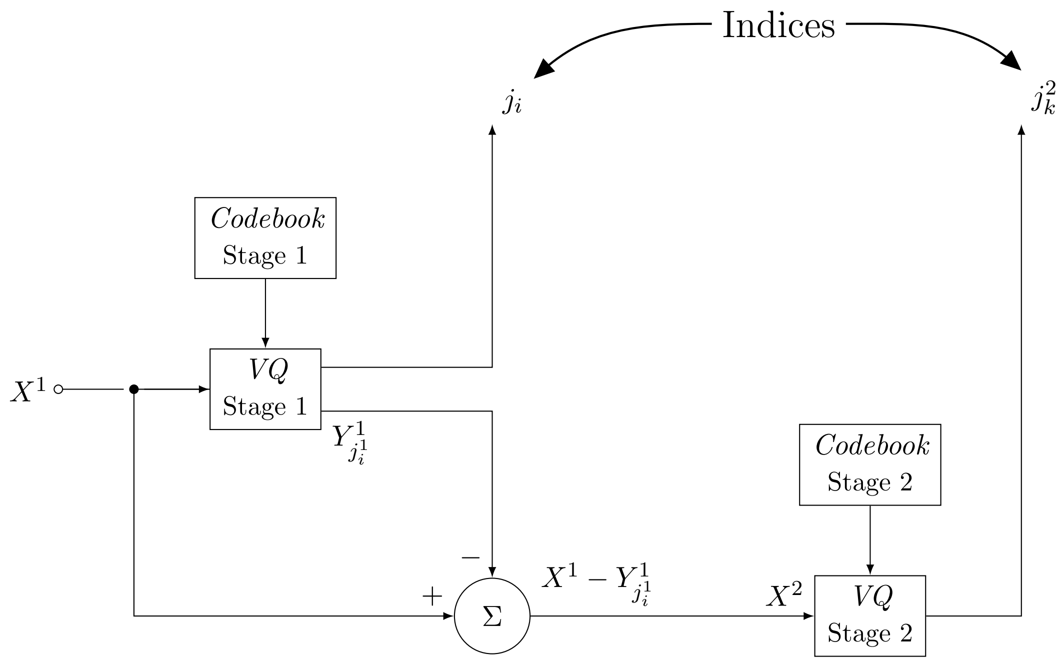

Fig. 1.2: Binary splitting (drawn with TikZ).

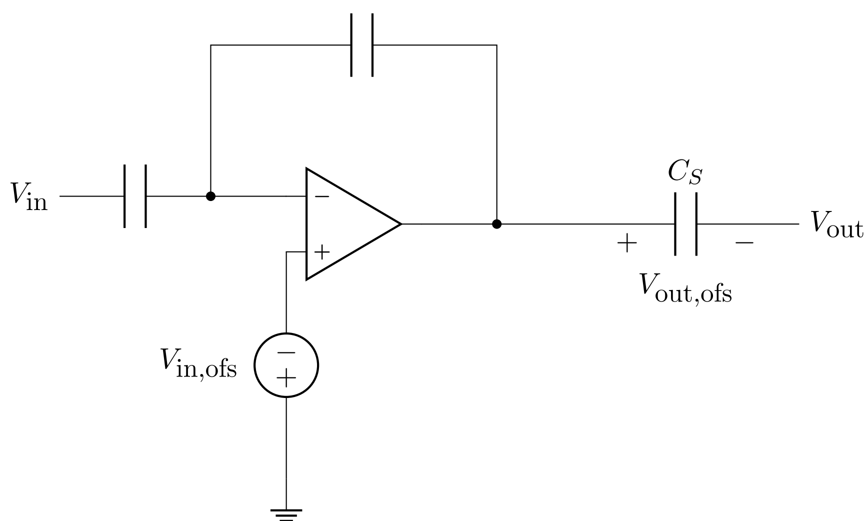

Fig. 1.3: Circuit example drawn using circuitikz.

There are many other ways to create figures. One package compatible with LaTeX is TikZ. An example is given in Fig. 1.2. This is identical to Fig. 1.1, except that it is done within the compiling process of LaTeX. Another example of a third-party figure package is given in Fig. 1.3. This circuit was generated using the `circuitikz` package.

It is important that there is no text between figures when they are referenced close together in the text. They should be "stacked" without text in between as seen above.

A final way of creating graphs is to use a open-sourse package called `PGFPlots`. An example of a good-looking graph generated using this package is given in Fig 1.4. Note that this figure is large enough that it is pushed by LaTeX to another page by itself and nicely centered.

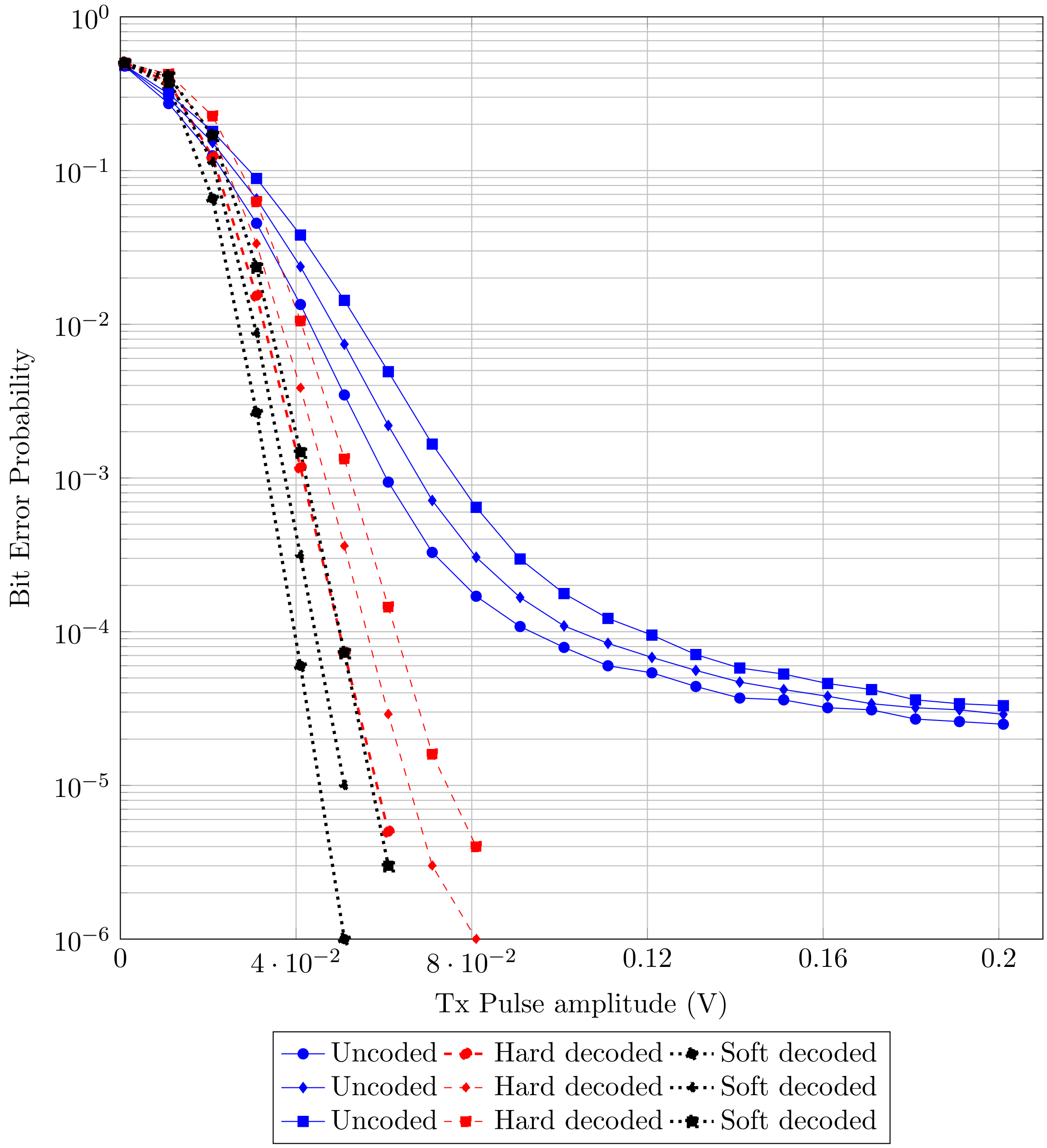

Fig. 1.4: Example figure made with PGFplots. Originally created in Matlab, then exported using the Matlab2TikZ script (available from Matlab Central). Then pasted into the LaTeX document and edited for style.

CHAPTER 2

RESIDUAL VECTOR QUANTIZATION AND ITS PROBLEMS

2.1 Residual Vector Quantization (RVQ)

A P-stage RVQ consists of a sequence of P single stage Vector Quantizers. Let us assume that the RVQ is made up of ESVQ stages. Each ESVQ is fully described by the set $\{ A^\rho, Q^\rho, P^\rho \}$. The method for designing the ESVQ is given in Algorithm 2.1. Note that this is the "usual" codebook design algorithm.

Throw in some citations [1–4].

Algorithm 2.1 LBG

Input:

    Training vectors (V_t),
    Distortion measurement rule d,
    Codebook size N,
    Threshold ε

Output:

    Codebook Vectors, Cb_i

Begin

    Select N initial codevectors, Cb_i
    Do
        Begin
            Partition V_t
            $Dist_{prev} = Dist_{current}$ /* Dist is the average */
            Calculate $Dist_{current}$ /* distortion of all the */
            Calculate Centroids of N groups of V_t /* training vectors when */
            Cb_i = Centroid of that group   /* partitioned or encoded */
        End
    while $\{(Dist_{prev} - Dist_{current})/Dist_{prev} \geq \varepsilon\}$

End

2.2 Reasons for the Poor Performance of RVQs

$$d(X^\rho, Y_i^\rho + A^{\rho+1} + \cdots + A^P) \leq d(X^\rho, A^\rho + A^{\rho+1} + \cdots + A^P) \tag{2.1}$$

It can be noticed from the above equation that while the traditional RVQ partitions are based on the stagewise residues, the optimal RVQ partitions are based on the final residues. As is evident from (2.1), the optimal codevectors are unique. The equivalent codevectors are obtained by summing all possible combinations of the codevectors of all stages. These represent the set of reconstruction vectors possible at the decoder.

2.3 Methods to Improve RVQ Performance

The various methods either suboptimal or optimal used in codebook generation and in the quantizer (RVQ) implementation are dealt with here. The common goal of all these methods is to improve the performance of the RVQ.

2.3.1 Brute Force RVQ or Stagewise RVQ (SRVQ)

The new codevectors are obtained by adding the centroids of the stagewise residues to the old codevectors. This can be done using a random splitting technique or a selective splitting technique.

2.3.2 Exhaustive Search RVQ (ESRVQ)

ESRVQ is the optimal RVQ described in the previous section. ESRVQ, as the name suggests, exhaustively searches all the *equivalent* codevectors as shown in (2.1). Centroids of the final residues are added to the codevectors during each iteration of the codebook design, to obtain the new optimal codevectors for the given partition.

2.3.3 Deep Search RVQ

Although ESRVQ is optimal, it needs an exhaustive search encoder. We must be able to create the encoder using an optimal method.

2.3.4 Comparison of SRVQ, DSRVQ, and ESRVQ Encoders

This section compares the different encoders presented previously. The different encoders have different performance and complexity, and so must be compared using a common basis. This is difficult to do, since we must first establish the criteria we will use.

2.3.5 Algorithm for Generating Jointly Optimized Codebooks

The ESRVQ is not instrumentable and the DSRVQ does not use a tree-structured encoder. Hence Barnes et al. proposed the reflection symmetric RVQ or the rRVQ [5]. The rRVQ uses a tree-structured encoder similar to SRVQ although it differs from the traditional RVQ or the SRVQ encoder in that it is slightly more complex. The rRVQ codebook is also more structured than the traditional RVQ.

Some other citations are in [3, 6–14].

2.3.6 Reflection Symmetric RVQ (rRVQ)

The ESRVQ is not instrumentable and the DSRVQ does not use a tree-structured encoder. Hence Barnes et al. proposed the reflection symmetric RVQ or the rRVQ [5]. The rRVQ uses a tree-structured encoder similar to SRVQ although it differs from the traditional RVQ or the SRVQ encoder in that it is slightly more complex. The rRVQ codebook is also more structured than the traditional RVQ.

The structured nature of the rRVQ codebook allows for a reduction of the complexity of the the implementation.

Binary rRVQ

It was already stated that for the optimal performance of the RVQ, an exhaustive search encoder must be used. To avoid this in rRVQ the codebook is constrained in such a way that the nearest neighbor stagewise equivalence classes are simply connected and convex [5]. A reflection symmetry is forced between the stagewise codevectors of the binary rRVQ to obviate the suboptimality caused by *entanglement* and *overlapping* discussed in the previous

Table 2.1: Performance results of ESRVQ, rRVQ, SRVQ, and DSRVQ of 4x4 vectors (PSNR in dB).

No. of		SRVQ		DSRVQ		ESRVQ		rRVQ	
Stages	bps	Unopt	JO	Initial	JO	Initial	JO	Initial	JO

chapter. Barnes et al. derived the optimality conditions for the rRVQ quantitatively [5]. They stated their results as follows [5, pp. 3–4]:

> “The difficulty in achieving optimality is that it is difficult. We observed that it was necessary to look at the conditions for optimality before we could proceed. We then proceeded with caution.
>
> Having proceeded, we applied the conditions for optimality. To our amazement, we found our results were optimal.”

2.3.7 Distortion Results and Analysis

Table 2.1 gives the PSNR in dB of the reconstructed test image, compressed (encoded and decoded) using the codebooks generated by SRVQ.

The ESRVQ is not instrumentable and the DSRVQ does not use a tree-structured encoder. Hence Barnes et al. proposed the reflection symmetric RVQ or the rRVQ [5]. The rRVQ uses a tree-structured encoder similar to SRVQ although it differs from the traditional RVQ or the SRVQ encoder in that it is slightly more complex. The rRVQ codebook is also more structured than the traditional RVQ.

It is important to recognize at this point, that rRVQ is a suboptimal method for covering the vector space. It is therefore important to make sure that the best possible vectors are chosen for the codebook.

REFERENCES

[1] B. Shucker, “A ground-based prototype of a CMOS navigational star camera for small satellite applications,” in *Proc. AIAA/Utah State University Conference on Small Satellites*, Aug. 2001, pp. 26–30.

[2] T. Brady, C. Tillier, R. Brown, A. Jimenez, and A. Kourepenis, “The inertial stellar compass: A new direction in spacecraft attitude determination,” in *Proc. AIAA/Utah State University Conference on Small Satellites*, Aug. 2002, pp. 21–25.

[3] T. K. Moon and W. C. Stirling, *Mathematical Methods and Algorithms for Signal Processing*. Upper Saddle River, NJ: Prentice Hall, 2000, ch. 13, pp. 591–620.

[4] I. Y. Bar-Itzhack and Y. Oshman, “Attitude determination from vector observations: Quaternion estimation,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-21, pp. 128–135, 1985.

[5] C. F. Barnes and R. L. Frost, “Residual vector quantizers with jointly optimized code books,” *Advances in Electronics and Electron Physics*, vol. 84, pp. 1–59, 1992.

[6] R. L. Baker and R. M. Gray, “Image compression using non-adaptive spatial vector quantization,” in *Proc. 16th Asilomar Conf. on Circuits, Systems and Computers*, Oct. 1982, pp. 55–61.

[7] C. F. Barnes and R. L. Frost, “Residual vector quantizers with jointly optimized code books,” in *Image Mathematics and Image Processing*, ser. Image Coding, P. W. Hawkes, Ed. Academic Press, 1993, pp. 57–69.

[8] C. F. Barnes, “Residual quantizers,” Ph.D. dissertation, Brigham Young University, Provo, UT, 1989.

[9] Analog Devices. (2000) 9851 DDS synthesizer. [Online]. Available: ftp://ftp.onsetcomp.com/Public/TattleTale/

[10] M. K. Ng, H. Shen, S. Chaudhuri, and A. C. Yau, “Zoom-based super-resolution reconstruction approach using prior total variation,” *Opt. Eng.*, vol. 46, no. 12, 127003 2007. [Online]. Available: http://dx.doi.org/10.1117/1.2818797

[11] B. L. Stringham, “RVQ gotchas,” Aug. 2000, Private Communication.

[12] T. C. Cournane and C. F. McSweeney, “Level measurement for storage silos,” U.S. Patent 4,807,471, Feb. 21, 1989.

[13] T. Berger, *Rate Distortion Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1971.

[14] C.-M. Huang, “Codebook generation for vector quantization image compression,” Master’s thesis, Utah State University, Logan, UT, 1988.

APPENDICES

APPENDIX A

List of Edge Vectors

A.1 Definition of an Edge Vector

Before we list the table of edge vectors, we need to describe what an edge vector is. In this section we will describe in detail the theory that results in the edge vectors. The first set of edge vectors is given in Table A.1.

A.2 Next Codebook Size Description

In this section we do the next size codebook. This is different from the previous case in that the codebook size is different. The next set of edge vectors is given in Table A.2.

A.3 Final Set of Codebook Size Descriptions

The following three tables contain the data for codebook sizes that are different than the previous sizes. We note that the differences in the tables are due to the differences in the sizes of the codebook edge vectors. Note the values given in Table A.3 – Table A.5.

Table A.1: List of edge vectors for a codebook with b=8 and d=3, for a 4×4 vector size.

Level	Edge Vectors
L1	(5)
	(6)
	(7)
L2	(3,1)
	(3,2)
	(3,5)
	(4,0)
	(4,2)
	(4,3)
	(4,4)
	(4,5)
	(4,6)
L3	(3,4,1)
	(3,4,2)
	(3,7,0)
	(3,7,2)
	(3,7,4)
	(4,1,0)
	(4,1,1)
	(4,1,2)
	(4,1,3)
	(4,1,4)
	(4,1,5)
	(4,1,6)

Table A.2: List of edge vectors for a codebook with b=4 and d=3, for a 4×4 vector size.

Level	Edge Vectors
L1	(1)
	(2)
	(3)
L2	(0,3)
L3	(0,2,0)
	(0,2,2)
	(0,2,3)

Table A.3: List of edge vectors for a codebook with b=16 and d=3, for a 4×4 vector size.

Level	Edge Vectors
L1	(11)
	(12)
	(13)
	(14)
	(15)
L2	(7,0)
	(7,1)
	(7,2)
	(7,6)
	(8,4)
	(8,5)
	(8,6)
	(9,6)
	(9,14)
	(10,1)
L3	(4,6,14)
	(5,6,6)
	(6,14,0)
	(6,14,3)
	(6,14,4)
	(6,14,5)
	(7,7,0)
	(7,14,7)
	(9,5,3)
	(9,5,10)
	(9,5,11)

Table A.4: List of edge vectors for a codebook with b=16 and d=3, for a 2×2 vector size.

Level	Edge Vectors
L1	(9)
	(10)
	(11)
	(12)
	(13)
L2	(6,0)
	(6,3)
L3	(2,2,8)
	(6,5,1)
	(6,5,4)
	(6,5,6)
	(6,5,7)
	(6,5,8)
	(6,5,15)
	(7,0,14)
	(8,0,1)
	(8,15,3)
	(8,15,4)
	(8,15,10)

Table A.5: List of edge vectors for a codebook with b=16 and d=3, for a 6×6 vector size.

Level	Edge Vectors
L1	(6)
	(7)
	(8)
	(9)
	(10)
	(11)
	(12)
	(13)
	(14)
	(15)
L2	(2,8)
	(2,13)
	(4,1)
	(4,6)
	(4,7)
	(4,8)
	(4,10)
	(4,11)
	(4,13)
	(4,15)
L3	(1,7,0)
	(1,7,1)
	(1,7,2)
	(1,7,3)
	(1,7,4)
	(1,7,6)
	(1,7,9)
	(1,7,12)

APPENDIX B

Another Example Appendix

B.1 Background

Some random appended text for this section of the appendix....

B.2 Meat of the Appendix

Here we have the data that is so important to be included in this appendix.

APPENDIX C

Example Appendix with Computer Code

```
#include "ISATLib.hch"

/***************************************************************************
 Macro Proc:          pipe_divide_uints

 Arguments
   x                  Dividend.
   y                  Divisor.
   val                Pointer to the result.
   fracBitsOut Number of bits in the fraction of the fixed-point quotient.

 Description
   Takes two signed integer inputs in any (non-Celoxica) fixed-point
   representation and finds their quotient. The number of fractional bits in
   the signed output is user specified. This is pipelined at one clock per pair
   with latency fracBitsOut+width(x)+2.

***************************************************************************/

macro proc
pipe_divide_uints(x,y,val,fracBitsOut)
{


  FLAG shift_sign[(width(x)+fracBitsOut)+1];
  unsigned (log2ceil(width(x)+1)+1) shift[(width(x)+fracBitsOut)+1];
  unsigned diff[(width(x)+fracBitsOut)+1];
```

```
unsigned divisor[(width(x)+fracBitsOut)+1];
unsigned quotient[(width(x)+fracBitsOut)+1];

int in_shifts;
unsigned in_divisor,in_divisor0,in_diff,in_diff0;
unsigned (log2ceil(width(x)+1)) msb_dividend,msb_divisor;

macro expr ext(p) = (int)((unsigned 1)0 @ p);

// Macro to find the number integer bits in the output. This macro produces an
// log2ceil(width(dividend1)+1) bit int.
macro expr int_shift(dividend1,divisor1) = ((dividend1 != 0)?
  (int)(lmo((unsigned 1)0 @ dividend1)) : (int)0) - (int)(lmo((unsigned 1)0 @
      divisor1));

par
{
  // Clock 0
  // Extendthe precision of the operands.
  in_divisor0 = y @ (unsigned (fracBitsOut))0;
  in_diff0 = x @ (unsigned (fracBitsOut))0;

  // Find shifts necessary to align MSBs.
  msb_dividend = (x != 0)? lmo(((unsigned 1)0 @ x)) : 0;
  msb_divisor = lmo(((unsigned 1)0 @ y));

  // Clock 1
  // Compute the total shift for the divisor to align MSBs.
  in_shifts = ext(msb_dividend) - ext(msb_divisor);
  in_divisor = in_divisor0;
  in_diff = in_diff0;
```

```
// Clocks 2 to (fracBitsOut+width(x)+2)
par(i=0 ; i <= (fracBitsOut+width(x)) ; i++){
  ifselect(i == 0){
    par
    {

      quotient[i] = 0;

      // Shift the divisor to align MSBs.
      if(in_shifts > 0){
        divisor[i] = in_divisor << (unsigned)in_shifts;
      } else {
        divisor[i] = in_divisor >> (unsigned) -in_shifts;
      }

      // Set the total number of shifts needed to find the quotient.
      shift[i] = (unsigned) (in_shifts + adjs((int)fracBitsOut,(log2ceil(width(
          x)+1)+1)));

      diff[i] = in_diff;
      shift_sign[i] = sign(in_shifts + adjs((int)fracBitsOut,(log2ceil(width(x)
          +1)+1)));
    }
  } else ifselect(i == (fracBitsOut+width(x))){

    if(shift_sign[i-1] == 0){

      // Find LSB of result.
      if((diff[i-1] >= divisor[i-1]) && (divisor[i-1] != 0)){
        *val = quotient[i-1] | 1;
      } else {
        *val = quotient[i-1];
```

```
    }
  } else

  // We are (effectively) dividing by zero; set the output to the dividend.
  *val = diff[i-1] @ 0;
  //*val = 0;

} else {

  if((shift[i-1] != 0) && (shift_sign[i-1] == 0)){
    par
    {

      if((diff[i-1] >= divisor[i-1]) && (divisor[i-1] != 0)){

        // Subtract off the shifted devisor and set an output bit.
        par
        {
          quotient[i] = (quotient[i-1] | 1) << 1;
          diff[i] = diff[i-1] - divisor[i-1];
        }
      } else {

        par
        {
          // Clear an output bit.
          quotient[i] = quotient[i-1] << 1;
          diff[i] = diff[i-1];
        }
      }

      divisor[i] = divisor[i-1] >> 1;
```

```
          shift[i] = shift[i-1] - 1;
          shift_sign[i] = shift_sign[i-1];
        }
      } else {

        // The quotent is computed; keep the values in the pipe.
        par
        {
          quotient[i] = quotient[i-1];
          diff[i] = diff[i-1];
          divisor[i] = divisor[i-1];
          shift[i] = shift[i-1];
          shift_sign[i] = shift_sign[i-1];
        }
      }
    }
  }
 }
}
```

CURRICULUM VITAE

John Q. Engineer

Published Journal Articles

- Rational Radial Distortion Models of Camera Lenses with Analytical Solution for Distortion Correction, Lili Ma, YangQuan Chen, and Kevin L. Moore, *International Journal of Information Acquisition*, *Accepted*.
- A Small Mobile Robot for Security and Inspection Operations, N.S. Flann, K. L. Moore, and Lili Ma, *Control Engineering Practice*, vol. 10, pp. 1265-1270, 2002.

Published Conference Papers

- Range Identification for Perspective Dynamic Systems Using Linear Approximation, Lili Ma, YangQuan Chen, and Kevin L. Moore, in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2004.
- Range Identification for Perspective Dynamic System with Single Homogeneous Observation, Lili Ma, YangQuan Chen, and Kevin L. Moore, in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2004.
- Blind Detection and Compensation of Camera Lens Geometrical Distortions, Lili Ma and YangQuan Chen, *SIAM Imaging Science*, 2004.
- Flexible Camera Calibration Using a New Analytical Radial Undistortion Formula with Application to Mobile Robot Localization, Lili Ma, YangQuan Chen, and Kevin L. Moore, in *Proc. Int. Symposium on Intelligent Control (ISIC)*, 2003.

- Sonar and Laser Based HIMM Map Building for Collision Avoidance for Mobile Robots, Lili Ma and Kevin L. Moore, in *Proc. International Symposium on Intelligent Control (ISIC)*, 2003.
- Wireless Visual Servoing for ODIS - An Under Car Inspection Mobile Robot, Lili Ma, Matthew Berkemeier, YangQuan Chen, Morgan Davidson, Vikas Bahl, and Kevin L. Moore, in *Proc. IFAC World Congress*, Spain, July, 2002.
- Visual Servoing of an Omni-Directional Mobile Robot for Alignment with Parking Lot Lines, Matthew Berkemeier, Morgan Davidson, Vikas Bahl, YangQuan Chen, and Lili Ma, in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, May 2002.
- Some Sensing and Perception Techniques for an Omnidirectional Ground Vehicle with a Laser Scanner, Zhen Song, YangQuan Chen, Lili Ma, and You Chung Chung, in *Proc. IEEE Int. Symposium on Intelligent Control (ISIC)*, October 2002.
- A Small Mobile Robot For Security and Inspection Operations, Flann NS, Moore KL, and Ma L, in *Proc. IFAC Conference on Telematics Applications in Automation and Robotics*, July 2001.