

Sobel Filter

Ray Seyfarth

June 29, 2012

Outline

- 1 Overview
- 2 Sobel in C
- 3 Sobel computed using SSE instructions

Overview

- The Sobel filter is an image processing edge detection algorithm
- It involves convolution of 3×3 image windows with 2 convolution matrices

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad S_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

- The edge value, G , for a pixel at (r, c) is computed by

$$G_x = \sum_{i=-1}^1 \sum_{j=-1}^1 (S_{x,i,j} * I_{r+i,c+i})$$

$$G_y = \sum_{i=-1}^1 \sum_{j=-1}^1 (S_{y,i,j} * I_{r+i,c+i})$$

$$G = \sqrt{G_x^2 + G_y^2}$$

A simple C solution

```
#include <math.h>
#define matrix(a,b,c) a[(b)*(cols)+(c)]
void sobel(unsigned char *data, float *output, long rows, long cols)
{
    int r, c;
    int gx, gy;
    for ( r = 1; r < rows-1; r++ ) {
        for ( c = 1; c < cols-1; c++ ) {
            gx = -matrix(data,r-1,c-1) + matrix(data,r-1,c+1) +
                -2*matrix(data,r,c-1) + 2*matrix(data,r,c+1) +
                -matrix(data,r+1,c-1) + matrix(data,r+1,c+1);
            gy = -matrix(data,r-1,c-1) - 2*matrix(data,r-1,c)
                - matrix(data,r-1,c+1) +
                matrix(data,r+1,c-1) + 2*matrix(data,r+1,c)
                + matrix(data,r+1,c+1);
            matrix(output,r,c) = sqrt((float)(gx)*(float)(gx)+
                                     (float)(gy)*(float)(gy));
        }
    }
}
```

Sobel using SSE instructions

- 16 8 bit values can be placed in an XMM registers
- The central 14 values can be used to compute 14 Sobel results
- The code loaded the row $r - 1$ and computed part of 14 Sobel results
- Then it loaded row r and added more to the 14 Sobel results
- Last it loaded row $r + 1$ and added more to the 14 Sobel results
- The contributions were added, squared, G_x^2 added to G_y^2 for 14 G values
- The 14 G values were written to the output image
- Using 1000 different images it processed 980 images per second vs 158 for the C code.
- This is 6.2 times as fast

New instructions used for Sobel

- `pxor` This instruction performs an exclusive or on a 128 XMM source register or memory and stores the result in the destination register.
- `movdqa` This instruction moves 128 bits of aligned data from memory to a register, from a register to memory, or from a register to a register.
- `movdqu` This instruction moves 128 bits of unaligned data from memory to a register, from a register to memory, or from a register to a register.
- `psrldq` This instruction shifts the destination XMM register right the number of bytes specified in the second immediate operand.
- `punpcklbw` This instruction unpacks the low 8 bytes of 2 XMM registers and intermingles them. I used this with the second register holding all 0 bytes to form 8 words in the destination.
- `punpckhbw` This instruction unpacks the upper 8 bytes of 2 XMM registers and intermingles them.

New instructions used for Sobel (2)

- `paddw` This instruction adds 8 16 bit integers from the second operand to the first operand. At least one of the operands must be an XMM register and one can be a memory field.
- `psubw` This instruction subtracts the second set of 8 16 bit integers from the first set.
- `pmullw` This instruction multiplies the first set of 8 16 bit integers times the second set and store the low order 16 bits of the products in the first operand.
- `punpcklwd` This instruction unpacks and interleaves words from the lower halves 2 XMM registers into the destination register.
- `punpckhwd` This instruction unpacks and interleaves words from the upper halves 2 XMM registers into the destination register.
- `cvtdq2ps` This instruction converts 4 double word integers into 4 double word floating point values.

The Sobel assembly code

- This code is far too long to examine in slides