Real-time image processing and object recognition for robotics applications

Adrian Stratulat
What is computer vision?

Computer vision is a field that includes methods for acquiring, processing, analyzing, and understanding images and, in general, high-dimensional data from the real world in order to produce numerical or symbolic information, e.g., in the forms of decisions.
What are its applications?

- OCR
- 3d scanning
- Obstacle avoidance
- Pattern recognition
- Object recognition
- Event detection
- Augmented reality
- Computer interaction
The Theory

Computer Vision
Color spaces

How do we represent colors as data?
Color spaces

RGB
(Red, Green, Blue)
Color spaces

**CMYK**
(Cyan, Magenta, Yellow, black)
Color spaces

HSV
(Hue, Saturation, Value)
Color spaces

HSL
(Hue, Saturation, Lightness)
Color spaces

Grayscale
Image processing

What can we do with images?
Simple filtering

Modifying the value of a pixel depending on its current value: tresholding, color space conversion
More advanced filtering (I)

Algorithms based on a convolution matrix:
Blurring, edge detection
More advanced filtering (II)

Sobel operator: finding edges:

$$G_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} \star A \quad \text{and} \quad G_y = \begin{bmatrix} +1 & +2 & +1 \\ +1 & +2 & +1 \\ +1 & +2 & +1 \end{bmatrix} \star A$$
More advanced filtering (III)

Scharr operator:

\[
G_x = \begin{bmatrix}
-3 & 0 & +3 \\
-10 & 0 & +10 \\
-3 & 0 & +3 \\
\end{bmatrix}
\]

\[
G_y = \begin{bmatrix}
-3 & -10 & -3 \\
0 & 0 & 0 \\
+3 & +10 & +3 \\
\end{bmatrix}
\]
More advanced filtering (IV)

Laplacian operator:

\[
\begin{bmatrix}
0 & 1 & 0 \\
1 & -4 & 1 \\
0 & 1 & 0 \\
\end{bmatrix}
\]

\[
dst = \Delta \text{src} = \frac{\partial^2 \text{src}}{\partial x^2} + \frac{\partial^2 \text{src}}{\partial y^2}
\]
More advanced filtering (V)

Smoothing (blurring):
- Gaussian blur
- Mean filtering
- Median filtering
- Bilateral filtering
From bitmaps to contours

After running edge detection, we can analyze the properties of the shape.
Advanced algorithms (I)

The Hough Transform can be used to detect even broken lines
The Hough Transform algorithm can also be modified to detect circles.
The Software: OpenCV

Open Source Computer Vision Library
OpenCV

- BSD licensed
- Written in C++
- Has bindings for C, C++, Java and Python
- Cross-platform
- Website: http://opencv.org/
OpenCV

• **Timeline:**
  - 1999 Intel started the project
  - 2000 first alpha release
  - 2006 version 1.0 launched
  - 2008 the project receives corporate support from Willow Garage
  - 2009 version 2.0 launched
OpenCV

- Modular design:
  - Core
  - Imgproc
  - Highgui
  - Calib3d
  - etc.
OpenCV

- Core:
  - Masking
  - Containers for data
  - etc.
OpenCV

- Imgproc
  - Blur
  - Tresholding
  - Structural analysis
  - Hough transform
OpenCV

• Highgui:
  - Windows
  - Trackbars
  - Reading and writing images
  - Camera capture
The Hardware: pcDuino

ARM Cortex-A8 dev board
pcDuino

Hardware specifications:

- Allwinner A10 SoC
- ARM Cortex-A8 core
- Mali400 GPU
- 1GB RAM
- 2GB Flash
- 3 USB ports
- 1 HDMI output
- 1 Ethernet port
pcDuino

CPU core details:

- 32 bit registers
- 31 general purpose registers
- ARMv7
- Superscalar architecture
- NEON SIMD unit
- VFP unit
pcDuino

CPU core – NEON unit:

- The NEON unit has 32 128bit registers for SIMD processing
- Supports arithmetic operations as 8bit, 16bit, 32bit signed and unsigned integers and simple-precision floating-point operations
- Operations include saturating arithmetic
pcDuino

CPU core – VPF unit:

- Supports arithmetic operations on single and double precision floating point numbers
- ISN'T A SIMD UNIT
The Optimizations

How to improve performance several times
Tips to improve performance on ARM (I)

Don't use bigger data types than you need!

Remember that the NEON registers are 128 bits wide, so in one clock cycle it can add 16 8-bit integers or 4 32-bit integers!
Tips to improve performance on ARM (II)

Avoid double-precision floating-point numbers!

This is the only data type that cannot be processed by the NEON unit, and will be processed by the VFP. (this isn't ideal as the VFP isn't a SIMD unit)
Tips to improve performance on ARM (III)

When iterating an array, prefetch!

ARM doesn't have automatic prefetching, so the “prefetch” assembly instruction needs to be called explicitly to avoid waiting for a cache miss later (prefetch brings the data from the RAM to the L2 cache).

If the compiler doesn't add it, modify the code.
Tips to improve performance on ARM (IV)

Don't be afraid to use ARM-specific instructions, like saturating arithmetic operations!

Even if you are afraid to write assembly, you may always use the processor's intrinsic functions.
Tips to improve performance on ARM (V)

Using some of these tips, I've managed to modify the OpenCV library to obtain speed-ups up to 6 times.
Questions ?