7. Image Processing

In this chapter we will do:

• Just a taste
• Auto-brightness
• Motion detection

Image Processing

• Getting an image from the sensor to the CPU is only the smallest part
• SOFTWARE is the key to make such a system useful!
• What are possible applications for “Embedded Vision Systems”?

Applications for “Embedded Vision Systems”

• Digital Camera
  store image data, display, upload
• Surveillance Camera
  upload, (store), (motion detection)
• Robotics
  object detection, motion detection, stereo vision, …

Software for “Embedded Vision Systems”

• Auto-Brightness
• Motion detection
• Color detection
• Shape detection

Which of these is
  – difficult to program?
  – compute intense?
7.1 Auto-Brightness

Idea for Auto-Brightness (gray scale image)
- Compute average by adding all gray values
- If average below threshold: open aperture  
  *(or digitally: increase all gray values by a factor)*
- If average above threshold: close aperture  
  *(or digitally: reduce all gray values by a factor)*

7.2 Motion Detection

Auto-Brightness Implementation (using CAMSet)

```c
void autobrightness(image img)
{
}
```

Idea for Motion Detection (gray scale image)

1. Subtract gray scale values for all pixel-pairs from two subsequent images

No Motion:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>0000</td>
<td>0000</td>
</tr>
</tbody>
</table>

Motion:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>0000</td>
<td>0000</td>
</tr>
</tbody>
</table>
```

- time t1
- time t2
Motion Detection

Idea for Motion Detection (continued)

1.a Better use diff. of absolute values than just difference
   • Compute average over all pixels
   • If average above threshold: motion detected!
     (returns 1 = motion, or 0 = no motion)

Motion Detection Implementation

```c
int motion(image im1, image im2, int threshold)
/* return 1 (true) if motion, 0 (false) if not */
{
    image diff;  
    int   avg;  
    difference(im1, im2, diff);  
    avg = average(diff);  
    return (avg > threshold);  
}
```

Motion Detection

Motion Detection Implementation

```c
int average(image im)
{
    int avg;
    // Calculation
    return avg;
}
```

7.3 Object Tracking

- We have done: check whether there is motion yes/no
- What would be nice: (assume camera is mounted on servo) Check **where** in the image motion is detected and move the camera towards it
- How can we do this ??

7.4 Color Object Detection

- Uniquely colored objects are easier to detect by using color than shape
- For a single **pixel**:
  - How can we check whether a pixel value matches a given sample color (RGB)
  - How about image noise, changing lighting conditions, HSV ?
- For a whole **image**:
  - How can we find the location of a colored object ?
Color Object Detection

Solution ideas

• Histograms for rows and columns
• Find max. value in histogram vs. weighted average of histogram

Image Processing

We have seen

• Image processing is not all that difficult if it is divided up into small functions
• This leads to the development of
  \( \Rightarrow \) Image Processing Libraries