Vision Based Autonomous Security Robot (VBASR)

Bradley University - ECE Department
Senior Capstone Project
Sponsored by Northrup Grumman
February 23, 2010
Student: Kevin Farney
Advisor: Dr. Joel Schipper
Presentation Outline

- What the project is...
- What has been completed so far...
- Where the project is going...
Project Summary

- What is VBASR?
  - Autonomous, Mobile, Security Camera

- VBASR is a computer vision project
  - Computer vision is defined as making useful decisions about real physical objects and scenes based on sensed images [1].

- Primary Goals – Using Computer Vision
  - Navigation
  - Obstacle Avoidance
OpenCV 2.0

- Intel
- Computer vision library
  - Image processing/computer vision
  - User interfacing
  - Core functionality
  - Machine learning
Image Processing

- Basic Ideas
  - Digital Signal Processing
  - Convolving Masks
  - Linear Algebra
- Image Properties
  - Grayscale, RGB, RGBA (alpha, opacity)
  - Bits per pixel
Vision Algorithm – Idea #1
Vision Algorithm – Idea #2
Vision Algorithm

1. **Camera**
   - **Current Frame**
   - **Smoothing Filter**
   - **Canny Edge Detection**

2. **Average Corner**
   - **Boundary Threshold**
   - **Corner Detection**
   - **Vertical Line Histogram**
   - **Strong Vertical Lines + Average**

3. **Decision Corners**
   - **Resolver**
   - **Decision Lines**

4. **Instruction**
Testing OpenCV - Filters

Normal Blur
Testing OpenCV - Filters
Testing OpenCV - Filters

Gaussian Blur
Testing OpenCV - Edge
Why Filters?

- Noise Reduction
Testing OpenCV - HighGui
Testing OpenCV - Corners
Selecting Parameter Values
Vision Algorithm - Example One

- Original Image to Edge Detection
Vision Algorithm - Example One

- Edge Detection to Corner Detection
Vision Algorithm - Example One

- Corner Detection to Vertical Lines
Vision Algorithm - Example One

- Corner Detection to Average Corners
Vision Algorithm - Example Two

- Original Image to Edge Detection
Vision Algorithm - Example Two

- Edge Detection to Corner Detection
Vision Algorithm - Example Two

- Corner Detection to Vertical Lines
Vision Algorithm - Example Two

- Corner Detection to Average Corners
Vision Algorithm - Example Two

![Image of a hallway with the word 'Straight' superimposed on it]
Next Steps

- Large Test Data Set Analysis
- OpenCV to control iRobot
  - Writing manifests within Microsoft Robotics Developers Studio (C#)
  - Utilizing OpenCV in C#
    - Wrapping my current functions (.cpp files)
    - Porting libraries and re-writing (.cs files)
- Netbook
Schedule – Spring Semester

<table>
<thead>
<tr>
<th>Milestone</th>
<th>Description</th>
<th>Expected</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>Edge Detection</td>
<td>26-Jan-10</td>
</tr>
<tr>
<td>10</td>
<td>Blob Tracking</td>
<td>2-Feb-10</td>
</tr>
<tr>
<td>11</td>
<td>Obstacle Detection</td>
<td>9-Feb-10</td>
</tr>
<tr>
<td>12</td>
<td>Navigation Vectors</td>
<td>16-Feb-10</td>
</tr>
<tr>
<td>13</td>
<td>Navigation Algorithms</td>
<td>22-Feb-10</td>
</tr>
<tr>
<td>14</td>
<td>Real-Time</td>
<td>2-Mar-10</td>
</tr>
<tr>
<td>15</td>
<td>Real-Time</td>
<td>9-Mar-10</td>
</tr>
<tr>
<td></td>
<td>Spring Break</td>
<td>16-Mar-10</td>
</tr>
<tr>
<td>16</td>
<td>Motion Detection</td>
<td>23-Mar-10</td>
</tr>
<tr>
<td>17</td>
<td>Motion Detection and Capture</td>
<td>30-Mar-10</td>
</tr>
<tr>
<td>18</td>
<td>Comparing and Recognizing Images</td>
<td>6-Apr-10</td>
</tr>
<tr>
<td>19</td>
<td>Mapping Hallway</td>
<td>13-Apr-10</td>
</tr>
<tr>
<td>20</td>
<td>Room Recognition</td>
<td>20-Apr-10</td>
</tr>
<tr>
<td>21</td>
<td>Complete Integration</td>
<td>27-Apr-10</td>
</tr>
<tr>
<td></td>
<td>Wrap up - Conference papers</td>
<td>4-May-10</td>
</tr>
<tr>
<td></td>
<td>Finals</td>
<td>11-May-10</td>
</tr>
</tbody>
</table>
Questions?

- VBASR by Kevin Farney
References


Filters - Normal

- Normal Blur
  - Normalized box filter – summation of pixels over a neighborhood

\[
K = \alpha \begin{bmatrix}
1 & 1 & 1 & \cdots & 1 & 1 \\
1 & 1 & 1 & \cdots & 1 & 1 \\
1 & 1 & 1 & \cdots & 1 & 1 \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots \\
1 & 1 & 1 & \cdots & 1 & 1
\end{bmatrix}
\]

\[
\alpha = \begin{cases} 
\frac{1}{\text{ksize.width}\times\text{ksize.height}} & \text{when normalize=true} \\
1 & \text{otherwise}
\end{cases}
\]
Filters – Gaussian

- Gaussian Blur
  - Convolution of source image with specified gaussian kernel

Matrix of ksize (parameter) x 1 with filter coefficients:

\[ G_i = \alpha \times e^{-\left(\frac{i-(\text{ksize}-1)/2)^2}{(2 \times \sigma)^2}\right), \]

\[ \alpha = \sum_i G_i = 1 \]
Filters

- Median Blur
  - Returns median of pixel neighborhood into the destination image for each pixel
Canny Edge Detection

- Implements Canny Algorithm
  - First noise-reduction needed (filters)
  - Intensity Gradients \( G = \sqrt{G_x^2 + G_y^2} \)
    - 8 points
  - Non-Maximum Suppression
  - Hysteresis Thresholding
    - High – discards noisy pixels
    - Low – connects the edges into lines (binary)
Corner Detection

- **Good Features To Track**
  - Calculates minimal eigenvalue per pixel
    - Covariation Matrix of derivatives
    - Then eigenvalues represent corners
  - Non-maxima suppression (3x3 pixels)
  - Rejection by quality level (parameter)
    - qualityLevel\*max(eigImage(x,y))
  - Rejection by distance (parameter)
Price Breakdown

- iRobot Create Premium Development Package
  - $299
- Pioneer 3-DX
  - upwards of $5000
- Microsoft Robotics Developers Studio R2
  - free download
- Visual Studio 2008
  - $500 and up
  - Visual C# editor – free download
- Small Netbook
  - Looking for around $300
Microsoft Robotics Developer Studio

- CCR (Concurrency and Coordination Runtime)
- DSS (Decentralized Software Services)
- VPL (Visual Programming Language)
- VSE (Visual Simulation Environment)