VBASR: The Vision System

Vision Based Autonomous Security Robot

Bradley University - ECE Department
Senior Capstone Project
Sponsored by Northrup Grumman
May 04, 2010
Student: Kevin Farney
Advisor: Dr. Joel Schipper
Presentation Outline

• What the project is…
• What has been completed…
• Results…
What is VBASR?
- Autonomous, Mobile, Security Camera

VBASR is a computer vision project

Primary Goals – Using Computer Vision
- Navigation
  - Obstacle Avoidance
Vision Algorithm

System Block Diagram

- VISION ALGORITHM
- IMAGE PROCESSING
- IMAGE STREAM
- CAMERA
- INSTRUCTIONS
- QUERY
- NAVIGATION ALGORITHM
- VBASE CONTROLS
- CONTROL SIGNALS
- FEEDBACK INFORMATION
- IROBOT DEVELOPMENT INTERFACE
- IROBOT CHASSIS
The Platform

- **Hardware**
  - iRobot Create
  - Webcam
- **Software**
  - OpenCV2.0
Vision Algorithm – Idea #1
Vision Algorithm – Idea #2
Vision Algorithm – Idea #3
Vision Algorithm – High Level

Diagram:
- Camera
  - Current Frame
    - Lines Algorithm
    - Corners Algorithm
    - Colors Algorithm
      - Resolver
        - Instruction
Vision Algorithm – Detailed
Feature Extraction

**Lines and Corners**

- Smoothing Filter
- Canny Edge Detection
- Corner Detection

**Colors**

Flood Fill
Testing OpenCV - Filters

Normal Blur
Testing OpenCV - Filters

Median Blur
Feature Extraction

**Lines and Corners**

- Smoothing Filter
- Canny Edge Detection
- Corner Detection

**Colors**

- Flood Fill
Testing OpenCV - Edge
Why Filters?

- Noise Reduction
Feature Extraction

LINES AND CORNERS

SMOOTHING FILTER → CANNY EDGE DETECTION → CORNER DETECTION

COLORS

FLOOD FILL
Testing OpenCV - Corners
Feature Extraction

**LINES AND CORNERS**
- Smoothing Filter
- Canny Edge Detection
- Corner Detection

**COLORS**
- Flood Fill
Testing OpenCV – Flood Fill
Vision Algorithm – Detailed

- Current Frame
  - Camera
  - Lines Algorithm
    - Corner Detection
      - Canny Edge Detection
      - Vertical Line Histogram
    - Smoothing Filter
      - Strong Vertical Lines + Average
      - Decision Lines
  - Colors Algorithm
    - Flood Fill
      - Evaluate Highest ‘Floor’ Pixels
        - Decision Colors
          - Decision Corners
  - Corners Algorithm
    - Additional Vertical Lines
      - Smoothing Filter
        - Canny Edge Detection
      - Corner Detection
  - Resolver
  - Instruction
Lines Algorithm

Feature Extraction

- Smoothing Filter
- Canny Edge Detection
- Corner Detection
- Vertical Line Histogram
- Strong Vertical Lines
- Average X-Value Projections
- Decision Lines
Lines Algorithm
Vision Algorithm – Detailed

Corners Algorithm

CURRENT FRAME

CAMERA

CORNER DETECTION

CANNY EDGE DETECTION

VERTICAL LINE HISTOGRAM

LINES ALGORITHM

COLORS ALGORITHM

CORNERS ALGORITHM

ADDITIONAL VERTICAL LINES

SMOOTHING FILTER

FLOOD FILL

EVALUATE HIGHEST ‘FLOOR’ PIXELS

DECISION LINES

DECISION COLORS

DECISION CORNERS

RESOLVER

INSTRUCTION
Corners Algorithm

Feature Extraction
Corners Algorithm
Colors Algorithm

Feature Extraction

COLORS ALGORITHM

FLOOD FILL

EVALUATE HIGHEST ‘FLOOR’ PIXELS

DECISION COLORS
Colors Algorithm
Vision Algorithm – Detailed
Vision Algorithm - Example One
Vision Algorithm - Example One
Vision Algorithm - Example One
Vision Algorithm - Example One
Vision Algorithm - Example One
Vision Algorithm - Example Two
Vision Algorithm - Example Two
Vision Algorithm - Example Two
Vision Algorithm - Example Two
Vision Algorithm - Example Two
## Quantitative Results

<table>
<thead>
<tr>
<th></th>
<th>Lines</th>
<th>Corners</th>
<th>Colors</th>
<th>Resolved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hard Left</td>
<td>33.3</td>
<td>13.3</td>
<td>93.3</td>
<td>93.3</td>
</tr>
<tr>
<td>Left</td>
<td>87.9</td>
<td>55.2</td>
<td>93.1</td>
<td>100.0</td>
</tr>
<tr>
<td>Slight Left</td>
<td>97.1</td>
<td>28.6</td>
<td>91.4</td>
<td>94.3</td>
</tr>
<tr>
<td>Straight</td>
<td>96.4</td>
<td>48.2</td>
<td>96.4</td>
<td>98.2</td>
</tr>
<tr>
<td>Slight Right</td>
<td>97.6</td>
<td>29.3</td>
<td>92.7</td>
<td>100.0</td>
</tr>
<tr>
<td>Right</td>
<td>57.1</td>
<td>46.9</td>
<td>96.9</td>
<td>95.9</td>
</tr>
<tr>
<td>Hard Right</td>
<td>26.3</td>
<td>21.1</td>
<td>100.0</td>
<td>94.7</td>
</tr>
<tr>
<td>Totals</td>
<td>70.8</td>
<td>34.7</td>
<td>94.8</td>
<td>96.6</td>
</tr>
</tbody>
</table>
Initial testing yields promising results!

- Two programs ran independently
  - Vision system
  - iRobot controls

- Verified quantitative results
- Exceeded expectations
Questions?

- VBASR by Kevin Farney


References


References

## Quantitative Results

<table>
<thead>
<tr>
<th></th>
<th>Lines</th>
<th>Corners</th>
<th>Colors</th>
<th>Resolved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hard Left</td>
<td>73.3</td>
<td>46.7</td>
<td>93.3</td>
<td>73.3</td>
</tr>
<tr>
<td>Left</td>
<td>79.3</td>
<td>89.7</td>
<td>93.1</td>
<td>98.3</td>
</tr>
<tr>
<td>Slight Left</td>
<td>100.0</td>
<td>68.6</td>
<td>91.4</td>
<td>97.1</td>
</tr>
<tr>
<td>Straight</td>
<td>100.0</td>
<td>69.6</td>
<td>96.4</td>
<td>100.0</td>
</tr>
<tr>
<td>Slight Right</td>
<td>97.6</td>
<td>51.2</td>
<td>92.7</td>
<td>100.0</td>
</tr>
<tr>
<td>Right</td>
<td>45.9</td>
<td>87.8</td>
<td>96.9</td>
<td>90.8</td>
</tr>
<tr>
<td>Hard Right</td>
<td>10.5</td>
<td>52.6</td>
<td>100.0</td>
<td>57.9</td>
</tr>
<tr>
<td>Totals</td>
<td>72.4</td>
<td>66.6</td>
<td>94.8</td>
<td>88.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Lines</th>
<th>Corners</th>
<th>Colors</th>
<th>Resolved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hard Left</td>
<td>33.3</td>
<td>13.3</td>
<td>93.3</td>
<td>93.3</td>
</tr>
<tr>
<td>Left</td>
<td>87.9</td>
<td>55.2</td>
<td>93.1</td>
<td>100.0</td>
</tr>
<tr>
<td>Slight Left</td>
<td>97.1</td>
<td>28.6</td>
<td>91.4</td>
<td>94.3</td>
</tr>
<tr>
<td>Straight</td>
<td>96.4</td>
<td>48.2</td>
<td>96.4</td>
<td>98.2</td>
</tr>
<tr>
<td>Slight Right</td>
<td>97.6</td>
<td>29.3</td>
<td>92.7</td>
<td>100.0</td>
</tr>
<tr>
<td>Right</td>
<td>57.1</td>
<td>46.9</td>
<td>96.9</td>
<td>95.9</td>
</tr>
<tr>
<td>Hard Right</td>
<td>26.3</td>
<td>21.1</td>
<td>100.0</td>
<td>94.7</td>
</tr>
<tr>
<td>Totals</td>
<td>70.8</td>
<td>34.7</td>
<td>94.8</td>
<td>96.6</td>
</tr>
</tbody>
</table>
Selecting Parameter Values

![Image of Edge Detection Window with sliders for Filter, High, and Low values]

![Image of Edge Detected + Corners window with sliders for Distance and Quality values]
Lines Algorithm - Problems
Corners Algorithm - Problems
Colors Algorithm - Problems
Colors Algorithm - Solution
Filters - Normal

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

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

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

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

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

\[ G_i = \alpha * e^{-(i-(ksize-1)/2)^2/(2*\sigma)^2}, \]

\[ \alpha = \sum_i G_i = 1 \]
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 $G = \sqrt{G_x^2 + G_y^2}$ $\Theta = \arctan\left(\frac{G_y}{G_x}\right)$
    - 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)