Automated Inspection

With

Machine Vision

Part 3

Stanley N. Hack, D.Sc., PE
ConsulTech Engineering, PLLC

www.consultechusa.com

November 7, 2016

Copyright © 2016. ConsulTech Engineering, PLLC. All rights reserved.
PRESENTATION GOALS

• Understanding *Machine Vision* and Its Uses
• Understanding *Machine Vision* Components
• Appreciating the Interacting Complexities of *Machine Vision* Components and Processing

Disclaimer:

• Most of Consul Tech Engineering’s clients regard their processes, which include *machine vision* applications, as highly proprietary.
• Many of the applications presented have been devised for this presentation.
PRESENTATION SCOPE

Part I Review: CNY Engineering Expo 2014
- *Machine Vision* Definition
- *Machine Vision* Uses
- *Machine Vision* Components – Illumination

Part II: CNY Engineering Expo 2015
- *Machine Vision* Components
  - Cameras and Sensors
  - Optics

Part III: CNY Engineering Expo 2016
- *Machine Vision* Processors
- *Machine Vision* Systems
- *Machine Vision* Software

Final Exam
**MACHINE VISION DEFINITION**

*Machine Vision* is defined as the technology and methods used to automatically inspect materials, components, and manufactured systems using image-based sensors and systems.

**Key Words**
- Automatic Inspection
- Materials, Components, Manufactured Systems
- Image-Based
MACHINE VISION DEFINITION

Machine Vision has the following attributes:

- The input is an image, and the output is a set of data, such as feature existence, object type, object or feature location, and measurements.
- The system analyzes inanimate objects.
- The system has *a priori* knowledge of the imaged object(s), including object shape, size, position, and attributes.
- The system performs its processing repeatedly and often rapidly.
- The imaging environment, including illumination, geometry, and motion, is controlled by the Machine Vision system.
Machine Vision is unique among all of the computer imaging modalities (computer vision, image processing, medical imaging, remote sensing) in that many aspects of the imaging environment can often be controlled:

- Illumination
- Sensor positioning
- Sensor size and resolution
- Image magnification and orientation
- Optical filtering
- Reflections

A large component of Machine Vision system design is the optimization of the captured images to make the software’s job feasible, easier, and/or faster.
USES OF MACHINE VISION

- Quality Control
- Process Control
- Robotic Guidance

Images courtesy of Microscan Systems, Inc., NERLITE® Lighting Solutions, Renton, WA

Image courtesy of Matrox Electronic Systems Ltd., Dorval, Quebec, Canada

NDA Expiration 01 December 2007

Copyright © 2016. ConsulTech Engineering, PLLC. All rights reserved.
MACHINE VISION COMPONENTS

- Cameras and Sensors
- Optics
- Illumination
- Synchronization
- Processing
- Reporting

Copyright © 2016. Consul Tech Engineering, PLLC. All rights reserved.
Automated Inspection with Machine Vision Part 1 summarized in two theorems:

**Theorem 1**
If adding shadows to a captured image helps the processing, configure the illumination to add the appropriate shadowing.

**Theorem 2**
If shadows in a captured image hurts the processing, configure the illumination to remove shadowing.

Figures courtesy of Illumination Technologies, Elbridge, NY
Review – Illumination

Imaging Geometry

- On-axis or coaxial
- Partial bright field or directional
- Diffuse, dome, or cloudy day
- Dark field
- Back lighting
- Structured

Review – Cameras and Sensors

- Interface
- Electronics
- Sensor
- Lens Mount
- Optics (Lens)

Back Focus

Copyright © 2016. Consul Tech Engineering, PLLC. All rights reserved.
## Review – Cameras and Sensors

### Sensor Types
- **CCD** – Charge-Coupled Device
- **CMOS** – Complimentary Metal Oxide Semiconductor
- **Bolometer** – measures the power of incident electromagnetic radiation via the heating of a material with a temperature-dependent electrical resistance

### Camera and Sensor Specification
- **Sensor Size**
- **Shutter Control**
- **Spectral Response**
- **Pixel Size**
- **Trigger Control**
- **Quantum Efficiency**
- **Frame Rate**
- **Scan Type**
- **Signal-to-Noise Ratio**
- **Sensor Resolution (Pixel Count)**

### Camera Interfaces
- **CameraLink**
- **CoaXPress**
- **USB Vision**
- **Analog (Legacy)**
- **GigE Vision**
- **Others (Legacy)**
# Review – Lens Specifications

- **Lens Type**
  - Finite / Finite Conjugates
  - Telecentric
- **Size (Diameter)**
- **Field-of-View (FOV)**
- **Focal Length (Snell’s Law)**
- **Depth-of-Field (DOF)**
- **Working Distance (WD)**
- **Magnification**
- **Numerical Aperture (f/Stop)**
- **Chromatic Optical Aberrations**
- **Resolution - Modulation Transfer Function (MTF)**
- **Contrast**
- **Lens Attachment**

![Lens Diagram](image_url)

*Courtesy of Edmund Optics, Barrington, NJ*
Machine Vision Processors

• Personal Computer (PC)
  o Inexpensive, fast, multiple sources
  o Intel Multi-Core architecture permits multi-processing
  o Packaging options – workstation, fanless, server

• Other Platforms
  o Graphical Processing Unit (GPU)
  o Field-Programmable Logic Array (FPGA)
  o Systems on a Chip (SoC)

• Smart Cameras

• Camera Interfaces (Frame Grabbers)
# Machine Vision Software Packages

<table>
<thead>
<tr>
<th>Type</th>
<th>Software</th>
<th>Company/Website</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Libraries</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HALCON</td>
<td>MVTec Software, GmbH, Munich, Germany, <a href="http://www.mvtec.com">www.mvtec.com</a></td>
<td></td>
</tr>
<tr>
<td>MIL</td>
<td>Matrox Electronic Systems Ltd., Dorval, Quebec, Canada, <a href="http://www.matrox.com">www.matrox.com</a></td>
<td></td>
</tr>
<tr>
<td>Sapera</td>
<td>Teledyne Dalsa, Inc., Waterloo, ON, Canada, <a href="http://www.teledynedalsa.com">www.teledynedalsa.com</a></td>
<td></td>
</tr>
<tr>
<td>OpenCV</td>
<td><a href="http://www.opencv.org">www.opencv.org</a></td>
<td></td>
</tr>
<tr>
<td>ImageJ</td>
<td><a href="https://imagej.nih.gov/ij/">https://imagej.nih.gov/ij/</a></td>
<td></td>
</tr>
<tr>
<td><strong>Graphical</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LabView</td>
<td>National Instruments Corporation, Austin, TX, <a href="http://www.ni.com">www.ni.com</a></td>
<td></td>
</tr>
<tr>
<td>MERLIC</td>
<td>MVTec Software, GmbH, Munich, Germany, <a href="http://www.mvtec.com">www.mvtec.com</a></td>
<td></td>
</tr>
<tr>
<td>Design Assistant</td>
<td>Matrox Electronic Systems Ltd., Dorval, Quebec, Canada, <a href="http://www.matrox.com">www.matrox.com</a></td>
<td></td>
</tr>
<tr>
<td><strong>Hybrid</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sherlock</td>
<td>Teledyne Dalsa, Inc., Waterloo, ON, Canada, <a href="http://www.teledynedalsa.com">www.teledynedalsa.com</a></td>
<td></td>
</tr>
<tr>
<td>Vision Editor</td>
<td>Keyence Corporation of America, Itasca, IL, <a href="http://www.keyence.com">www.keyence.com</a></td>
<td></td>
</tr>
</tbody>
</table>
Machine Vision Software Packages

Matrox Design Assistant

- Select the action to perform from a context-based list
- Conveniently switch between the flowchart and image, or operator views
- Get quick access to context sensitive help
- Configure each step without losing sight of flowchart and image
- Instantly view results after each step
- Track and navigate the flowchart execution history without losing sight of the image
Machine Vision Software Packages

Dalsa
Sherlock
Machine Vision Software

• **Image Acquisition** – Grab image(s) from camera(s)
  o Synchronize with material handling system
  o Synchronize with or control illumination (optional)

• **Image Pre-Processing**
  o Point Processes – Gray-level adjustments
  o Arithmetic – Combination of two or more images
  o Spatial Filters – Convolution and Morphology
  o Geometric Transforms – Alignment

• **Feature Segmentation** – Separate features of interest from background

• **Feature Analysis**
  o Measure and compare to specifications
  o Pattern matching
  o Locate artifacts (debris, flaws, missing features, ...)

• **Reporting**
  o Control material handling
  o Communicate to operators and servers
  o Generate logs
Software – Image Representation

Pixel Values

black \hspace{1cm} gray \hspace{1cm} white

0 \hspace{1cm} \hspace{1cm} \hspace{1cm} N - 1

Pixel values:

- black: 0
- gray: 1 to N - 2
- white: N - 1

Note:

Pixel(i, j) where:

- 0 \leq i < m
- 0 \leq j < n
Software – Point Preprocessing

- Contrast stretching
  - Min / Max
  - Mean / Standard Deviation
- Histogram Equalization
- Cutoff
- Saturate
- Threshold – create binary image

Point Process Applications
- For visualization of images taken under different lighting conditions – Example: Stitching together a mosaic of image for remote sensing
- Normalization of images when automatic comparison operations are used

\[ g(i,j) = F(f(i,j)) \]
A gray-level histogram is a plot showing the number of pixels containing each gray-level in an image.
Software – Point Preprocessing

Min/Max Contrast Stretching

![Histogram Comparison](image_url)

Original

Stretched – Move and Combine Histogram Bins

Copyright © 2016. ConsulTech Engineering, PLLC. All rights reserved.
Software – Point Preprocessing

Histogram Equalization

Original

Equalized to Create Uniform Distribution

Original Cumulative

Equalized Cumulative
Software – Point Preprocessing

Note bright reflections at tops of pins

Original Image

Histogram Stretch

Histogram Equalization

Original Image

Histogram Stretch

Histogram Equalization
Software – Point Preprocessing

Original Image – Red = Dark Image
Blue = Light Image

Histogram Equalized – Red = Dark Image
Blue = Light Image

Original Image – Histogram Difference
(Light Image – Dark Image)

Histogram Equalized – Histogram Difference
(Light – Dark)
Software – Arithmetic Preprocessing

Pixel-By-Pixel Arithmetic, Logic, or Masking

\[ g(x, y) = F( f_1(x, y), f_1(x, y) ) \]

\[ g(x, y) = F( f_1(x, y), C) \]
Software – Arithmetic Preprocessing

• Pixel-By-Pixel Masking

• **Pixel-By-Pixel Subtraction** – a *poor* way to perform pattern matching
Software – Arithmetic Preprocessing

• Pixel-By-Pixel Adding

Original RGB Color Image

Red Component

Green Component

Blue Component

\[ \sum \]

Computed Gray Image
Software – Filter Preprocessing

Convolution Filter:

\[ g(i,j) = \left( (f(i-1,j-1) \times w_0) + (f(i,j-1) \times w_1) + (f(i+1,j-1) \times w_2) + \\
( f(i-1,j) \times w_3) + (f(i,j) \times w_4) + (f(i+1,j) \times w_5) + \\
(f(i-1,j+1) \times w_6) + (f(i,j+1) \times w_7) + (f(i+1,j+1) \times w_8) \right) \div C \]

For all \(0 \leq i < m\) and \(0 \leq j < n\)
Software – Filter Preprocessing

Convolution Filter:

Average

<table>
<thead>
<tr>
<th>1/9</th>
<th>1/9</th>
<th>1/9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/9</td>
<td>1/9</td>
<td>1/9</td>
</tr>
<tr>
<td>1/9</td>
<td>1/9</td>
<td>1/9</td>
</tr>
</tbody>
</table>

Edge

<table>
<thead>
<tr>
<th>-1</th>
<th>-1</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>9</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>
Software – Filter Preprocessing

Convolution Filter:

North/South Edge

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>-1</td>
</tr>
</tbody>
</table>

East/West Edge

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>
# Software – Filter Preprocessing

**Convolution Filter:**

**Original Image**

**Sobel Edge – 4-Connected**

<table>
<thead>
<tr>
<th>0</th>
<th>2</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>-2</td>
<td>0</td>
</tr>
</tbody>
</table>

+  

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>-2</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Sobel Edge – 8-Connected**

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>-2</td>
<td>-1</td>
</tr>
</tbody>
</table>

+  

<table>
<thead>
<tr>
<th>1</th>
<th>0</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>-2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>-1</td>
</tr>
</tbody>
</table>
Binary Morphology:

**Dilate**

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

4-Connected

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

8-Connected

Change center pixel from 0 to 1 if any X = 1

**Erode**

<table>
<thead>
<tr>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

4-Connected

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

8-Connected

Change center pixel from 1 to 0 if any X = 0

Thresholded Image

Dilate 3 Iterations

Erode 3 Iterations
Software – Filter Preprocessing

Binary Morphology:

**Open** – Multiple erode operations followed by multiple dilate operations

**Close** – Multiple dilate operations followed by multiple erode operations
**Software – Geometric Transformation**

**Translation**
Requires one fiducial point
\[ g(x, y) = f(x + t_x, y + t_y) \]

**+ Magnification**
Requires two fiducial points
\[ g(x, y) = f(m_x(x + x'), m_y(y + y')) \]

**+ Rotation**
Requires three fiducial points
\[ g(x, y) = a \cdot x^2 + b \cdot y^2 + c \cdot x \cdot y + d \cdot x + e \cdot y + f \]
Software – Feature Segmentation

Example: Back lit hardware

Backlit Image

Thresholded Image

Gray-Level Histogram

Threshold = 63
Software – Feature Segmentation

Example: Vials are filled with a liquid. Are they filled the proper amount?

Not Filled  Partially Filled  Properly Filled

- Illumination: Back lighting
- Camera: 640 x 480 area scan
- Optics: 15 mm c-mount lens
Software – Feature Segmentation

Properly Filled Example

Invert Gray Levels

Sobel Edge Detection

East-West Directional Edge Detection
Software – Feature Segmentation

Partially Filled Example

Invert Gray Levels

Sobel Edge Detection

East-West Directional Edge Detection
Software – Feature Segmentation

Empty Example

Invert Gray Levels

Sobel Edge Detection

East-West Directional Edge Detection
# Software – Feature Analysis

## Hough Transform

<table>
<thead>
<tr>
<th>Step</th>
<th>Find Lines</th>
<th>Find Circles</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equation</strong></td>
<td>$x \cos \theta + y \sin \theta = r$</td>
<td>$x^2 + y^2 = r^2$</td>
</tr>
<tr>
<td><strong>Parameter Space</strong></td>
<td>$r, \theta$</td>
<td>$x, y$ for each $r$ (one plane for each $r$)</td>
</tr>
</tbody>
</table>
| **Parameter Space**   | 1. Add the gray-level of each pixel in the image space $(x, y)$ where $r = \sqrt{x^2 - y^2}$ and $\theta = \cos^{-1} (x/r)$ to the parameter space at coordinates $r, \theta$  
  2. Threshold parameter space | 1. Add the gray-level of each pixel in the image space whose $x, y$ coordinates solve the above equation for the selected value of $r$ to the parameter space at coordinates $x, y$  
  2. Threshold the parameter space |
| **Image Space**       | Draw lines in the image space for all points in parameter space $(r, \theta,)$ whose values exceed the threshold where $x = \cos(r)$ and $y = \sin(r)$ | Draw circles in the image space with centers at $(x,y)$ and with radius $r$ for all points in parameter space that exceed the threshold |
Software – Feature Analysis

Hough Transform

Find Lines

Find Circles

Image Space

Hough Space

Image Space

Hough Space for Radius = r

Copyright © 2016. ConsulTech Engineering, PLLC. All rights reserved.
Software – Feature Analysis

Projections

Extract a 1-D slice of pixels from a 2-D image

Average pixel values in direction perpendicular to projection line
## Software – Feature Analysis Example

<table>
<thead>
<tr>
<th></th>
<th>Back Lighted Image</th>
<th>Inverted Gray Levels</th>
<th>East/West Edge Detection</th>
<th>Hough Line Detection</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Empty Vial</strong></td>
<td><img src="image1.png" alt="Empty Vial Image" /></td>
<td><img src="image2.png" alt="Inverted Empty Vial Image" /></td>
<td><img src="image3.png" alt="East/West Edge Detection" /></td>
<td><img src="image4.png" alt="Hough Line Detection" /></td>
</tr>
<tr>
<td><strong>Full Vial</strong></td>
<td><img src="image5.png" alt="Full Vial Image" /></td>
<td><img src="image6.png" alt="Inverted Full Vial Image" /></td>
<td><img src="image7.png" alt="East/West Edge Detection" /></td>
<td><img src="image8.png" alt="Hough Line Detection" /></td>
</tr>
<tr>
<td><strong>Partially Filled Vial</strong></td>
<td><img src="image9.png" alt="Partially Filled Vial Image" /></td>
<td><img src="image10.png" alt="Inverted Partially Filled Vial Image" /></td>
<td><img src="image11.png" alt="East/West Edge Detection" /></td>
<td><img src="image12.png" alt="Hough Line Detection" /></td>
</tr>
</tbody>
</table>
## Software – Feature Analysis Example

<table>
<thead>
<tr>
<th>Inverted Gray Levels and Projections</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Empty Vial</strong></td>
</tr>
<tr>
<td><img src="image1.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="graph1.png" alt="Graph" /></td>
</tr>
<tr>
<td><strong>Full Vial</strong></td>
</tr>
<tr>
<td><img src="image2.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="graph2.png" alt="Graph" /></td>
</tr>
<tr>
<td><strong>Partially Filled Vial</strong></td>
</tr>
<tr>
<td><img src="image3.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="graph3.png" alt="Graph" /></td>
</tr>
</tbody>
</table>
Software – Feature Analysis Example

<table>
<thead>
<tr>
<th>Empty Vial</th>
<th>East/West Detection and Projections</th>
<th>Hough Line Detection and Projections</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="image1" alt="Empty Vial East/West Detection" /></td>
<td><img src="image2" alt="Empty Vial Hough Line Detection" /></td>
</tr>
<tr>
<td></td>
<td><img src="image3" alt="Empty Vial Projections" /></td>
<td><img src="image4" alt="Empty Vial Projections" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Full Vial</th>
<th>East/West Detection and Projections</th>
<th>Hough Line Detection and Projections</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="image5" alt="Full Vial East/West Detection" /></td>
<td><img src="image6" alt="Full Vial Hough Line Detection" /></td>
</tr>
<tr>
<td></td>
<td><img src="image7" alt="Full Vial Projections" /></td>
<td><img src="image8" alt="Full Vial Projections" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Partially Filled Vial</th>
<th>East/West Detection and Projections</th>
<th>Hough Line Detection and Projections</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="image9" alt="Partially Filled Vial East/West Detection" /></td>
<td><img src="image10" alt="Partially Filled Vial Hough Line Detection" /></td>
</tr>
<tr>
<td></td>
<td><img src="image11" alt="Partially Filled Vial Projections" /></td>
<td><img src="image12" alt="Partially Filled Vial Projections" /></td>
</tr>
</tbody>
</table>
Software – Feature Analysis

Blob (Binary Large Object) Analysis:

1. Performed on binary image
2. Find first “white” pixel
3. Recursively grow blob in all directions from the white pixel
4. Find next white pixel that is not part of a blob
5. Repeat steps 3 and 4 until entire image is analyzed
6. Compute metrics:
   a. Ellipse Perimeter
   b. Ellipse Area
   c. Ellipse Major and Minor Radii
   d. Ellipse Rotation
   e. Enclosing Rectangle
   f. Circularity = perimeter² / (4π area)
Software – Feature Analysis

Blob Analysis:

Backlit Image

Threshold

Open

Backlit Image Histogram

Blob Analysis

Blob Analysis
Software – Feature Analysis

Pattern Matching:

1. Find a segment of a search image, \( f( i, j ) \), that matches the contents of a template image, \( t( i, j ) \) where the template image is smaller than the search image.

2. Uses:
   - Match fiducial points or segments for image alignment
   - Presence/Absence analysis

3. Operation:
   a. Slide template over search image and calculate Normalized Cross-Correlation at each increment of sliding \((m, n)\)
   b. Find peak value of Normalized Cross-Correlation optionally by fitting a curve (bi-linear, bi-quadratic, ...) to the NCC map near the peak value

\[
NCC = \sum_{i,j} \left[ (f( i, j ) - \bar{f} ) \ast ( t( i + m, j + n ) - \bar{t} ) \right]
\]

where:
- \( m_{min} \leq m \leq m_{max} \)
- \( n_{min} \leq n \leq n_{max} \)

\[
NCC = \frac{ \sqrt{ \sum_{i,j} (f( i, j ) - \bar{f} )^2 } \ast \sum_{i,j} ( t( i + m, j + n ) - \bar{t} )^2 }{ \sum_{i,j} (f( i, j ) - \bar{f} )^2 \ast \sum_{i,j} ( t( i + m, j + n ) - \bar{t} )^2 }
\]

for: \( 0 \leq NCC \leq 1.0 \)
Software – Feature Analysis

Pattern Matching: 5x5 Template, 7x7 Image

Normalized Cross-Correlation Scores
Software – Feature Analysis

Pattern Matching to Analyze Presence/Absence:

<table>
<thead>
<tr>
<th>Object</th>
<th>Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good</td>
<td>Score = 100</td>
</tr>
<tr>
<td>Reversed</td>
<td>Score = 34</td>
</tr>
<tr>
<td>Missing Label</td>
<td>Score = 0</td>
</tr>
<tr>
<td>Partial Label</td>
<td>Score = 59</td>
</tr>
</tbody>
</table>

Copyright © 2016. Consul Tech Engineering, PLLC. All rights reserved.
Software – Feature Analysis

Pattern Matching to Analyze Presence/Absence:

Object

Template

Score = 90

Score = 85

Score = 76

Dark Image

Histogram Equalized

Label Misplaced

Object Rotated

Score = 57

Score = 90

Score = 76

Copyright © 2016. Consul Tech Engineering, PLLC. All rights reserved.
Software – Feature Analysis

**Pattern Matching:**

**Object**

**Mask – East/West Edge Detection**

**Match – East/West Edge Detection**

Score = 100

Small Template

Match - Small Template

Score = 100

---

**Strategies to reduce processing time:**

- Use as small a template as possible
- Use thresholded binary image if possible
- Preprocess to remove detail if possible
- Use *pyramid* scheme
Example – Semiconductor Wafers

- Locate wafers as they are extracted from an oven.
- Determine if any wafers are broken.
- Send location of unbroken wafers to robot for extraction from the rack.
- Report any broken wafers.
Example – Semiconductor Wafers

- **Goal** – Illuminate and image **only** the wafer edges
- **Illumination** – Back lighting
- **Camera** – Line Scan
Example – Semiconductor Wafers

- Capture line scans at fixed time intervals
- Threshold each line
- Analyze multiple vertical profiles across the image width to locate wafers
- Analyze Profiles:
  - Calibrate dimensions from *a priori* information about wafer spacing
  - Determine wafer condition
  - Determine wafer location
- When the entire rack has been extracted, send coordinates of undamaged wafers to robot for extraction
Example

TSA Training System
Simulated Threat Weapons

Challenges:
- 1980’s time frame
- Compensate for moving belt
- Processing speed
- Threat weapon must be totally enclosed by luggage
SUMMARY

- **Machine Vision** is defined as the technology and methods used to automatically inspect materials, components, and manufactured systems using image-based sensors and systems.

- **Machine Vision** is used for **Quality Control**, **Process Control**, and **Robotic Guidance**.

- **Machine Vision Systems** include the following components:
  - Illuminators
  - Cameras and Sensors
  - Optics (Lenses)
  - Synchronization (Motion Control)
  - Processors

- **Machine Vision Software** is able to extract, highlight, and manipulate information from a captured image, but it cannot add information. In other words, *if the software can’t see it, it can’t process it.*

- **Machine Vision Systems Engineering** is required for success.
1. **What is *Machine Vision*?**
   *Machine Vision* is the technology and methods used to automatically inspect materials, components, and manufactured systems using image-based sensors and systems.

2. **What are the uses of *Machine Vision*?**
   *Machine Vision* is used for Quality Control, Process Control, and Robotic Guidance.

3. **What are the components of a *Machine Vision System*?**
   A *Machine Vision System* includes illuminators, sensors (cameras), optics (lenses), synchronizers (motion controllers and/or encoders), processors, and software.

4. **Name some of the *Machine Vision* processing platforms?**
   PC, smart cameras, GPU, FPGA, Systems on a Chip (SoC).

5. **What are the five Machine Vision processing steps?**
   1. Image Acquisition
   2. Image Pre-Processing
   3. Feature Segmentation
   4. Feature Analysis
   5. Reporting
Thank You For Attending

Automated Inspection
With
Machine Vision
Part 3

Stanley N. Hack, D.Sc., PE

ConsulTech
ConsulTech Engineering, PLLC
www.consultechusa.com

November 7, 2016