Applying Machine Vision to Verification and Testing

Ben Dawson and Simon Melikian
ipd, a division of Coreco Imaging, Inc.
www.goipd.com

Abstract

Machine vision is a superior replacement for human vision in applications such as high-speed verification and testing. We review trends and issues in applying machine vision. In the last few years, machine vision systems have improved dramatically in performance, ease-of-use, intelligence, and cost. Applying machine vision to verification and testing is now relatively easy and inexpensive. Successful application of machine vision requires care in the selection of the vision vendor, the vision system, and other components in your system, and in integrating the vision system into your production process.

Introducing Machine Vision

Machine vision uses cameras, computers and algorithms to replace human vision in inspection tasks that require precise, repetitive, high-speed verification and testing. Humans are not good at making precise measures “by eye” and their performance quickly decreases when doing repetitive or high-speed visual tasks.

On the other hand, the human visual system is unmatched for understanding complex visual scenes, quickly learning new visual tasks, and making subtle judgments based on limited evidence. Machine vision still must be limited to verifying and testing well-specified parts in controlled environments.

As an example of the differences in human and machine capability, Figure 1 shows a dot-matrix barcode. A defect is a missing dot, which could be caused by two dots touching each other or a dot with the wrong size. Can you see the defect? How many dots are there? What are the dot areas in microns^2?

![Figure 1. Dot inspection – a good task for a machine vision system](image)

The defect is 29 bars from the right, where top two dots are touching. There are over 1300 dots, so counting them is slow and difficult. Precise dimensional measure of each dot is impossible for us to do – our visual system is not a calibrated measuring tool – although we can point out dots that touch or look too big or too small.
You could do these boring tasks using an optical gauging system but it would take you minutes. The barcodes come by every 0.6 seconds. This is just the kind of demanding, repetitive testing and verification task where machine vision excels and makes economic sense.

In Figure 2, you can easily point out the trees and other objects and could guide your car on the road (unless you are a Boston driver).

Figure 2. You understand this kind of natural scene; a machine doesn’t

These kinds of general vision tasks are currently too complex and variable to be reliably and economically done by machine vision. We expect that machine vision will get more capable over time and replace human vision in a wider range of tasks. For example, there have been rapid advances in recognizing human faces using a machine vision system.

Your first question in applying machine vision, should be, “Is this an appropriate and economical application for machine vision?” Is the vision task well specified, quantitative, precise, fast or repetitive?

Your second question should be, “Can a particular machine vision system do the application?” Does the machine vision system have the computational “horsepower,” appropriate software, and algorithms to do what you want it to do?
A trusted vendor or integrator can help you answer these questions. The vendor or integrator can often demonstrate a solution “on the spot,” using an integrated vision system with Rapid Application Development (RAD) software tools. If not, you will have to work with the vendor or integrator to solve the application. Be aware that most vision applications are not as well defined as you might initially think. You should be prepared to spend additional time and money as you discover issues and accommodate changes in the application.

**What is a Machine Vision System?**

Some machine vision system vendors consider a machine vision system to be only image capture and processing hardware and software. We prefer to call this the “vision system” (VS) and see it as the key component in a larger “machine vision system” (MVS). The terminology may vary, but what is important is that the vision system cannot work alone; it must be part of and integrated into a larger system that, in turn, is integrated into your production.

Figure 3 is a schematic diagram of a typical machine vision system. In this example, we are measuring the threads on screw-top bottles.

![Figure 3. Components in a typical machine vision system](image)

The major components of this machine vision system are:

- A fixture or staging to position the part in front of a camera. A conveyor belt, in this example, positions bottles in front of the camera.
- A “Part-in-Place” (PiP) sensor tells the vision system when a part (a bottle) is available for testing and verification.
• Lighting to illuminate the part. Proper lighting is critical for MVS success!
• A camera and lens acquires an image of the part (the bottle’s neck).
• The vision system (VS) has hardware to put the camera image into one or more processors, software (algorithms) to perform the test and verification, and a way to communicate the results of these tasks.
• Communication of the results. In this example, we signal a “kicker” to put defective bottles onto another conveyor for recycling.

You can find these components in very different types of MVS. In web inspection, for example, the MVS inspects a continuous flow of material such as paper or plastic. In this case the material is “fixed” by rollers and the PiP might be a photocell to signal when the material starts and ends. Image acquisition is continuous using a line-scan camera clocked by an encoder. Lighting for web inspection is often demanding and expensive because of the large width across the web. The vision system might require additional processors to keep up with the data stream from the cameras. The consequences of a defect could be a tag on the material or a “roll map” listing the distances to defective areas.

Your third question in applying machine vision should be to consider what other components are needed to make a system, the physical layout of the system, and what integration into the existing product line is required. For example, does your production line need to be reworked to accommodate the lighting and cameras? You need to consider the total MVS cost, not just the cost of the vision system.

Trends in Vision Systems

In the last few years, vision systems have improved dramatically in performance, ease-of-use, intelligence, and cost. Many more applications can now be economically solved with machine vision.

Trends in Performance

Vision system vendors started using commodity x86 and DSP processors in the early 1980’s. Processor performance has increased dramatically since then and, therefore, so has vision system performance. In 1984 a convolution (a standard machine vision operation) took 50 seconds on a VAX 11/750. The same operation on a Pentium™ 4 processor is now under 5 milliseconds, a speed-up of over 10,000 times.

Commodity processors have the additional, major, benefit that a large body of standard software and programmer knowledge can be leveraged to develop vision software. By re-using their existing software “components,” vendors can focus on developing new capabilities for their vision systems.
Vision system performance has also improved due to better algorithms – the procedures and methods used to solve a machine vision application. Algorithms are implemented both in hardware and software but we consider only software implementations, except when talking about high performance systems.

Vision algorithms have become more robust and better able to handle variations in the image and adapt to changes. They have become much better at precise, accurate and repeatable measurements. They also have become “smarter,” for example, automatically learning what to verify instead of having to be programmed. Smarter software might not decrease testing time, but certainly decreases the application development time.

Other components of a machine vision system have also improved. For example, a “framegrabber” is specialized hardware that transfers images from the camera into the processor’s memory. The trends are to integrate the framegrabber into either the camera or the vision processor, or to add performance and capabilities to the framegrabber for demanding vision applications.

Today’s machine vision frame grabbers have specialized features such as Trigger-to-Image Reliability™ – the ability to sense the PiP (Part in Place) signal, acquire the image, and transfer the image to the processor with low latency and high reliability. Other features include the ability to work with special machine vision cameras and high data transfer rates.

*Trends in Ease-of-Use and Intelligence*

Machine vision is a complex technology and so can be difficult to apply. Ease-of-use is very important because it manages the complexity and difficulties so you can quickly solve your application. We will argue that ease-of-use – including better user interfaces, smarter vision systems, and easier integration – is currently the major factor driving vision system development.

The first machine vision software came from university and research labs in the early 1970’s. By the 1980’s, there were established methods for basic operations and work shifted to developing algorithms for such things as precise measurement, optical character recognition (OCR), and automatic search. These algorithms are now well known, but better versions and newer algorithms continue to be developed as they provide competitive advantages for a machine vision vendor.

At first there was little concern with the ease-of-use of a machine vision system. The focus was on getting the fundamentals working! The software was packaged in function libraries, a format that is still used. Interpreters and point-and-click selection of functions made it a little easier for users to string together lower-level operations into a complete vision application.
Vendors realized that they were losing or delaying sales by not having easy-to-use software. Attempts in the 1980’s to build user-friendly products failed because the processors of the time did not have the performance to do the vision computation and support the graphics and intelligence needed to make the products easy to use.

In the 1990’s, ease-of-use made rapid advances due to improvements in Microsoft Windows™, faster processors, and better paradigms for the user interface. Coreco Imaging created two rapid application development (RAD) packages that can cut software development time for an application from months to days. Using the WiT™ package, an application is developed by graphically connecting functions into a “data flow” diagram. The Sherlock™ package uses an image-operation paradigm. You develop your application by graphically selecting image areas to test and verify, and then selecting operations to apply to these image area (see Figure 4).

![Sherlock Image](image.png)

*Figure 4. Sherlock uses an image-operation to rapidly develop vision applications*

To further improve ease-of-use, most vendors now offer small, integrated vision systems (IVSs), perhaps with limited or fixed functions. The terminology used by vendors for their IVSs is confusing. For example, one vendor’s “smart camera” might be equivalent to another vendor’s “intelligent sensor.” Until this is sorted out, we describe four general types of IVSs.
All IVSs integrate the components of a vision system (VS): processor, software, and communications. This integration and the IVS’s small size, makes them easy to integrate with your manufacturing process.

Some IVSs include the camera and a few include lighting. Integrating the camera makes a smaller package but also limits what the IVS can do. Lighting is very application specific, so it makes little sense to integrate lighting into the IVS.

Our general types of IVSs are, in order of increasing size and capability:

**Vision sensors** integrate a camera, frame grabber, processor, and communication into a small package. They are low in cost and easy to use, but extremely limited in what they can do.

**Smart Cameras** integrate a camera and vision system into a slightly larger package. The processor is programmable and so the ease-of-use depends on the software design. Most commonly the software is a RAD package – typically based a set of algorithm “tools”.

**Vision Appliances™** (a term unique to ipd) are like Smart Cameras (see Figure 5a), with a plug-in camera. The significant difference is the new type of software used on the Appliances.

![Figure 5a](image1.png) A Vision Appliance (about 6" x 3" x 3"). 5b A NetSight II Compact Vision System (not to the same scale as 5a – the NetSight II is about 3 times the size of a Vision Appliance)

**Compact Vision Systems** are very capable vision systems in compact packages. Our example is the NetSight II (see Figure 5b). The NetSight II runs the Sherlock RAD package but it can also run the Appliance software and the Appliance can be a target for an application developed by Sherlock.

The key points are, first, you can choose from a range of size and performance points to best meet your application’s requirements. Second, these IVSs are easy to install and integrate into your process. Third, Vision Appliances
have a new type of software that takes ease-of-use for application development to a new level.

We found that there are many potential users of machine vision who are not vision experts and have fairly standard applications. For example, a plant engineer might want to use vision to improve his / her manufacturing process or monitor a trouble spot on a production line. These users don’t have the time, interest, or direction to master machine vision or even a RAD package, and their applications generally don’t require the power of a RAD package.

The Vision Appliance software was developed to meet this need. Here are some of the ideas (not all unique) that went into the Appliance software:

First, each Appliance is designed for a specific range of applications, such as measurement (iGauge), printed label inspection (iLabel) or product assembly and verification (iCheck). We can then build in specific knowledge about the application, so that you don’t have to add this knowledge, as you would with RAD software. iGauge “knows” about measurements – distances, angles, tolerances, etc. iLabel “knows” about labels – position, rotation, flags, splatters, etc. and iCheck “knows” about product assembly and correctness – presence/absence, surface flaw detection, counting, etc. The Appliances are prepackaged vision solutions.

Second, the user doesn’t understand vision, so this knowledge is “in the box”. We picked the algorithms and added “expert systems” to set their parameters. Unlike a RAD package, you never mess with algorithms.

Third, there is no programming. The Appliances learn by example. To train iLabel, you show it good and bad labels and it automatically learns to identify good and bad labels. There are parameters to set, but we are careful to present these parameters to you in familiar terms, such as “position tolerance” rather than “edge detection”.

Fourth, the user interface is attractive, easy, and simple. We looked at the existing user interface paradigms, including our own, and decided that they were all too complex and technical for an Appliance user. No surprise – most of these packages were written by machine vision engineers or students, and have more power and quirks than most users want.

So we started with a clean screen and designed a new interface. We used ideas from our existing products, but carefully presented in a non-technical way. We used the “best practices” in user interface design and had the results reviewed by a professional GUI designer, who was not familiar with machine vision (see Figure 6). The result is a user interface that you will be comfortable with in a few minutes.
Figure 7. A Vision Appliance screen, from the iCheck™ Appliance.

**Trends in Cost**

Vision systems have decreased in cost, following the cost of processors and other parts. Smart Sensors are inexpensive, but the cost of more capable vision systems has not come down dramatically. You are paying somewhat less for hardware but getting huge improvements in performance and ease-of-use.

When you factor in the reduction in development, integration and set-up time because of improved ease-of-use, the decrease in cost is more dramatic. As an example, Table 1 compares the implementation of a barcode task, similar to the one described above, if done in 1996 and 2004:

<table>
<thead>
<tr>
<th>Measure</th>
<th>1996 Estimate</th>
<th>2004 Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vision System</td>
<td>ITI Series 150/40</td>
<td>ipd NetSight II</td>
</tr>
<tr>
<td>Cost (+ development)</td>
<td>$10,000 (+$11,000)</td>
<td>$4,000 (+1,000)</td>
</tr>
<tr>
<td>Time to Plant Floor</td>
<td>4 months</td>
<td>1 month</td>
</tr>
<tr>
<td>Size, in cubic inches</td>
<td>2736</td>
<td>216</td>
</tr>
<tr>
<td>Power in watts</td>
<td>500</td>
<td>72</td>
</tr>
</tbody>
</table>

Table 1. Comparing a barcode quality application. First cost estimate is for the hardware, the next estimate is the additional cost of development. All values are only for the vision system.
Looking Forward

Speculations on technology trends often make embarrassing reading a few years later. They can also prevent sales while customers wait for the “better” solution that might not happen. So assuming you won’t tease us or wait for an uncertain future, here are our speculations on where vision systems are going.

Processor performance will continue to improve but not at the rates we have seen. Most vision algorithms are memory intensive and memory speed has not improved as fast as processor speed. For example, doubling processor speed no longer halves the application’s cycle time. Instead we might see only 20% or 30% gains, due to memory speed limitations.

There will be some performance increases as more parallel instructions, such as Intel’s MMX, are added to general-purpose processors, but much of this gain has been “tapped out” already. Special “media processors” used in consumer goods are not economical in the much smaller machine vision system market.

Improving algorithms will make machine vision systems “smarter” (and use more processor power). Smarter vision systems are more robust and easier to use.

Vision systems cost will continue to decrease, but not dramatically. There will be inexpensive vision systems for large markets, such as security systems, but most machine vision applications are too small to ride the economies of scale.

Frame grabbers and attached processors will improve as “accelerators” for high-performance vision applications. We see the increasing use of programmable hardware (e.g., FPGAs) for high-performance vision tasks.

We expect substantial growth in small, integrated vision systems with easy-to-use software and ease-of-integration hardware. These systems hit the “sweet spot” for performance and cost and, more importantly, make it possible for a non-vision expert to use machine vision for their applications.

As vision systems get smarter and faster, they will take on more complex visual tasks. As more vision systems are sold, the vendors can afford to make them smarter and cheaper. Someday there might be a vision system smart enough to drive us around Boston – or smart enough to not want to drive us around Boston.
Issues in Applying Machine Vision

We discussed applications suitable for machine vision, and some questions to ask when considering applying machine vision. We reviewed the components of a machine vision system and trends in vision systems. Applying machine vision is much easier than it was a few years ago due to improvements in ease-of-use, integration and performance. If your application requires simple measurements, defect detection, alignment (part finding), etc., then an integrated vision system (IVS) will probably be your choice.

We now look further at four issues in the successful application of machine vision: software development, performance, lighting, and choosing a vendor.

Software Development

Software development was, along with lighting and integration, a major cost in putting together a machine vision system. The introduction of RAD (rapid application development) packages greatly reduced software development time and costs, but still requires some level of expertise. Where an Appliance will do, the software development is done.

Getting comfortable with a vision RAD package and doing your application takes a few days to a few weeks. There are two “gotchas” to watch out for. First, some RADs seem simple to use but get more difficult as you get into them. We suspect these RADs are a “bag” of tools written by different authors and without consistent design specifications. Second, are the capabilities and the algorithms in the RAD sufficient for your application? The authors had to limit what is in the RAD to make it easier to use. Many vendors will let you try their RAD packages, and we recommend doing so.

If the chosen RAD proves unusable or unable to do what you want, you can give up or get help from the vendor. At some point, the vendor will charge for helping. We think, perhaps cynically, that some vendors price their hardware, training, and RAD tools very attractively knowing that you will have to pay them for engineering services to actually get the application done. Talk to other customers to get their experience with the RAD package and ask the vendor for a schedule of support fees.

If you have a specialized application, need new algorithms, or need better performance, you can “go deep” with the function libraries, such as Sapera™, that underlie the RAD packages. This makes sense in an OEM application where the increased software development time is repaid by better performance and lower unit cost over many units.
The Performance Divide

There are vision tasks where the data rates or computation required will overwhelm a general-purpose x86 or DSP processor and “desk top” operating systems. For these high-performance applications you will need additional, dedicated machine vision processing hardware. This hardware is typically on the frame grabber or an attached processor. In any case, you are no longer talking about an integrated vision system; rather you are now integrating your own vision system or paying someone to do it.

An example of this kind of application is web inspection. Suppose we need 2 mil (50 micron) resolution on an 8-foot wide web moving 300 feet per minute. Our raw data rate is 120 megabytes per second, sustained. As of this writing, this is beyond what a standard processor can do. You might use an advanced frame grabber designed for the special “line scan” camera used, coupled with an attached processor such as the Coreco Imaging Anaconda™.

Some RAD packages support some hardware “acceleration”, but you often need to use the vendor’s libraries and write C or C++ code. This is not too bad, compared to the cost and time for the other components of a high-performance vision system.

We can’t give an exact point or divide where hardware acceleration is recommended. As an approximation, applications with sustained data rates of greater than 10 megabytes per second – roughly 30 parts per second – need some level of hardware assistance. It depends strongly on what operations are being done. The point (in our favor) is to pick a vendor that can supply the acceleration, if needed.

Successful application of machine vision requires care in the selection of the vision vendor, the vision system, and other components in your system, and in integrating the vision system into your production process.

Lights, Action, Camera

One of the more challenging aspects of developing a machine vision system (MVS) is getting a good image. A good image clearly shows what you want the vision system to “see.” This requires positioning of the part, careful lighting, and the proper camera. If, for example, you are looking at scratches on shiny metal parts, the lighting needs to be “low angle” with respect to the part and the camera perpendicular (normal) to the surface of the part. Otherwise, all you will see is the reflection of the lights off of the parts.

Lighting, in particular, is both a science and somewhat of an art. There are many companies that make specialized lights for machine vision, and most vendors either recommend “qualified” vendors or re-sell their lighting. A thick
catalog of lights, however, doesn’t tell you which one to use. Many of the lighting companies will be glad to recommend lighting if you send them a part to evaluate. Both the lighting and machine vision vendors have information on their web sites. There are also courses given by the lighting vendors and the machine vision vendor that will at least cover the fundamentals in their training courses.

After receiving recommendations from the lighting and machine vision vendors, be prepared to get some lights and experiment. Sometimes the ideal lighting is not possible because of physical constraints on the manufacturing line. Often you can’t get the lighting to really show what you need to see, so the results from the machine vision system will be less reliable.

You should plan for some time and cost to set up good lighting. The challenges and uniqueness of each lighting situation is why we don’t integrate lighting with our integrated vision systems.

Choosing a Machine Vision Vendor

Choosing the right vendor is obviously important to the success of your application of machine vision. Here are some general pointers to consider.

First is a bit of due-diligence: Are you comfortable that the vendor will be around when you need them, in 2 or 5 or 10 years? Of course no one really can know, but you can get some comfort by looking at the vendor’s history. How long have they been in business? What is their financial situation? Is this their main business or a "sideline" that might be dropped in a corporate power struggle? Do they have a serious commitment to research and new product development?

Second, how are they on support? Can you use a product, like an Appliance, that has very little support requirements? Even with modern RAD tools, you will need some training and development support. Does the vendor include some development support costs in their product price or is that a hidden cost you will discover later? How long does the vendor maintain a product line? What does a maintenance contract cost? We are not saying that support should be free or that you should expect the vendor to be able to guess how much support you will need. Rather, try to understand the costs and factor them into the total cost of ownership.

Third, does the vendor have more powerful products that, if needed, you could move your application to? As we have discussed, at some point an IVS or frame grabber in a personal computer runs out of processing “bandwidth”. This divide point is often not clear until you are over it. We hope that the vendor will “right size” your application from the start, but some insurance never hurts.