ROBOTIQ

Vision Systems for Collaborative Robots

Start Production Faster

Vision Systems for Collaborative Robots

LEAN ROBOTICS

robotiq.com | leanrobotics.org

Table of Contents

Lean Robotics: Simplify Robotic Cell Deployments

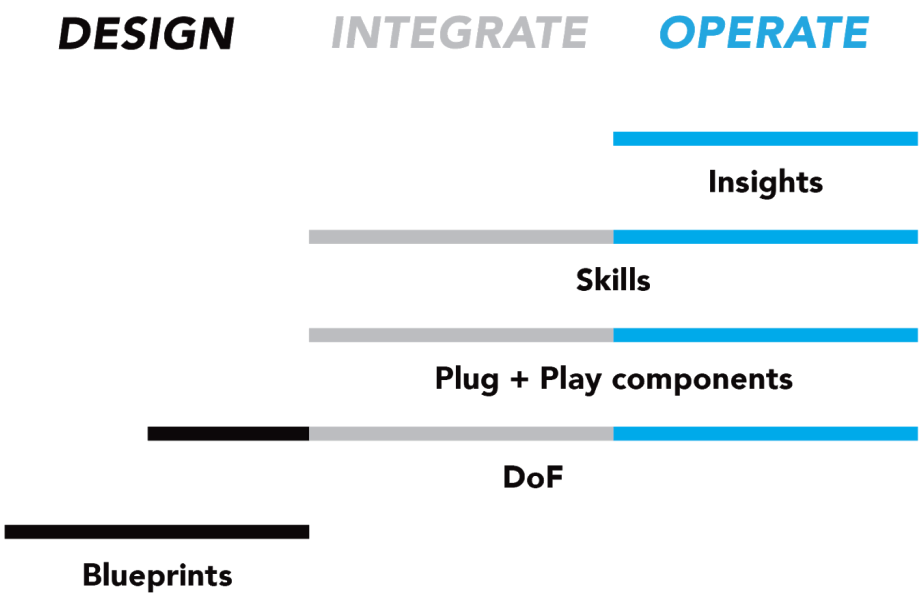
If you ask whether robots could work in your factory, the answer you'll get is probably a hesitant "It depends." It depends on your factory, your team, which robot you choose, what you want it to do... and a whole lot more.

So if you're a first-time robot user, how can you get started? How do you get from your initial idea to a productive, working robot? And if you've already got a few robotic deployments under your belt, how can you scale up your robotics efforts throughout your factory – or across multiple factories?

The answers can be found in **lean robotics: a methodology for simplifying robotic cell deployments**.

Lean robotics is a systematic way to complete the robotic cell deployment cycle, from design to integration and operation. It will empower your team to deploy robots quicker and more efficiently than ever before.

Lean robotics divides robotic cell deployments into three phases: Design, Integrate, and Operate.



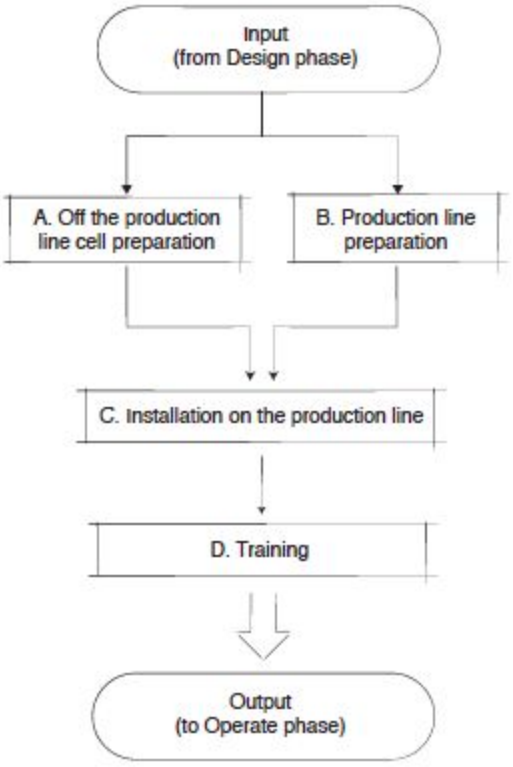
Robotiq's library of eBooks covers each phase of robotic cell deployment, giving you access to advice from robotics experts each step of the way.

Learn more about Lean Robotics at **leanrobotics.org**.

This eBook Covers the Integrate Phase

The integrate phase consists of putting the pieces of the robotic cell together, programming, and installing the cell on the production line.

Before starting the integrate phase, you should have the cell design in hand and the equipment ready to be assembled. At the end of the integrate phase, you'll have a working robotic cell on your production line that's ready to start creating value for its customer.



Introduction

Maybe you're thinking about integrating vision into your current system or your next automation project. Or maybe you've just bought a vision system and need to know how it works. If any of the above applies to you, keep reading!

You'll see there are a lot of different vision systems out there, which incorporate a variety of technologies for a range of applications. So it's important to acquire some basic knowledge of machine vision to help you figure out exactly what a simple vision application is.

It might also be useful to understand the differences between this simple type of application and a more complex one. Whether you're just starting with vision, or wondering whether adding vision to your system might be a solution for you, this eBook is a great place to start.

What Can You Do with a Vision System?

Not so long ago, most robots were blind. But today, adding a sense of vision to your robot can open up a world of possibilities. Think of everything you do with your eyes, and you'll start seeing the possibilities: part inspection, bin picking, counting, sorting by color... There are lots of applications out there, so it's critical to evaluate your project up-front to make sure you select the right technology for the job.

There are three different aspects of a vision system that determine whether it's simple or complex.

The first aspect is how images are captured. The idea behind image capture is to enhance the features you want to see and dampen everything else.

Although it sounds simple, this can be quite an art form: it all depends on your application and setup. Working with highly reflective metallic parts, for example, can be quite challenging.

You will need to rely on techniques used in photography to control the lighting intensity and camera position in order to achieve a satisfactory result.

The second aspect concerns image analysis. If the images were captured correctly, you should have a sharp contrast between the background and the object (e.g., a black rectangular part lying on a backlit white surface). In this case, you'll be able to use simple image analysis tools to recognize the shape of the object.

On the other hand, if for instance you need to find a target in a 3D space where you have many objects in the camera's field of view, your application will be much more complicated.

So consistent shapes with clear-cut boundaries and sharply contrasting backgrounds will be easy to analyze. By contrast, shapes that vary from one part to another, or shapes placed on low-contrast backgrounds may require more advanced vision systems.

In the latter case, you'll need more advanced knowledge of vision systems and image analysis.

Image capture and image analysis are directly related. Sometimes it's worth working harder on image capturing in order to have an easier time with image analysis. However, if there are variations you can't handle at the capturing stage, more advanced image analysis tools might be able to help you out.

The third aspect is what you want to do with the vision data. That is, once your system has located the part and identified its coordinates, what do you want to do with this information?

Simple vision projects may only establish a pass/fail status determining whether or not the part has been found. There are many tools out there that will allow you to set up this kind of application

with minimal effort.

However, if you want to send the data to the robot so the robot can pick up the part, that's a totally different story. In this type of application there are a lot more variables to work out.

First, you need to calibrate the camera frame, so the robot will know where the camera is and where it is looking. Then, you need to calibrate the plane of the object that you want to locate. Finally, you need to convert the positions of the objects in the camera's coordinate system to positions that can be read by the robot's coordinate system.

If you haven't done linear algebra in a while, this is prime territory for developing headaches. Instead, you might want to use tools that are easy to integrate with a robot, like the Robotiq Wrist Camera. An integrated system will take care of all of this for you, and you won't need any linear algebra skills to set up the application.

Vision System Applications and Alternatives

This section covers the most popular types of applications for vision systems. If you're not sure whether a vision system is right for you, we've also listed alternatives that might do the job.

Pick & Place

When you're deciding whether to use a vision system or another option for your pick and place operation, the most important factor is usually part presentation. You need to locate the parts in order to let the robot know where to pick them up.

We often focus on pick position, but the same factors apply to *place* position. Depending on what you need to do when placing your part, you may also need a vision system to determine the proper placement (e.g., locating the box in which the part is to be placed).

Here are some part localization options for pick and place applications:

- **Fixtures**—This option refers to placing parts in trays or securing them with fixtures. Fixtures ensure that parts are always at a constant position relative to the robot. This makes programming much simpler, since all you need to do is teach the positions to the robot.

  Some robot models come with palletizing programs that will help you teach positions faster. If you already have this kind of setup, it's probably overkill to add vision. But if you don't have such a system, consider the following: designing a fixture can be costly (depending on your parts).

Also, fixture design can be tricky if you want to feed a lot of parts to the robot without recharging the jigs. In this case, you need to be able to place as many parts as possible within the robot's reach.

Another downside to fixtures is if you have several different parts for the robot to pick. For this scenario, you may need several different jigs, which will add to the cost.

You should also consider changeover time when you're estimating the efficiency of the system. If your robot always manages the same kind of part, or only a few different ones, then fixtures might be the way to go. However, a vision system will probably give you more flexibility if you have to manage different parts with the same automated system.

- **Bowl Feeders**—These are meant to take bulk items and singularize them, so parts are presented to the robot one at a time, and the robot always picks the part from the same position.

  You only have to program one pick position, and once a setup like this is installed and adjusted, it usually works well.

  However, this type of equipment can be quite expensive, and depending on the shape of the parts you're picking, adjusting it can be finicky. Plus, bowl feeders usually limit you to sorting only one kind of part. You'll have to add another bowl feeding system to your robotic cell if you want to switch between different parts.

- **Conveyors**—Conveyors can be a good option for part presentation. However, if you want to use a conveyor without a vision system, you need to have a datum line or point of reference that your part will rest against (so parts are always picked from the same position).

  In this case, you'll probably still need a presence sensor to let the robot know when the part has arrived at the picking location. Some robots can pick parts while the conveyor is moving, but if your robot lacks this option, you might have to stop the conveyor during the picking application and restart it once it's done.

  If you don't have a datum line, or the parts aren't always in the same position or at the same spot on the conveyor, you'll probably need a camera to locate parts on the conveyor.

- **Vision Systems**—This option is typically used to find a part's location and orientation.

  There are many ways to use a vision system for part recognition:

  a. Use a 2D camera to locate parts resting on a stable surface.
  b. Use a fixed or robot-mounted camera to locate parts on a moving conveyor.
  c. Have the robot take a 3D scan of a surface and search for parts.

Every option has pros and cons. If you need more in-depth guidance on how vision might apply to or improve your application, get in touch with an expert – they'll help you out.

Vision is a good option if you're switching parts often. You can usually teach a robot to pick new parts pretty quickly, and unlike with jigs, there are no additional hardware costs. In general, you just need to adjust the vision algorithm and you're good to go.

You can also save the different algorithms so they're easy to reuse later. Instead of taking up space in your plant to stock different jigs, you simply store the vision algorithms on your PC or in your smart camera.

The price for such a system will vary depending on the technology. It's important to get the right technology for your needs, which is why we recommend talking to experts before buying. But remember, you don't want to go overboard on the specs and end up buying something expensive that you won't fully use!

Quality Control

Quality control is another major factor that can be improved by automation. First, you must identify which aspects of the part to look at in order to say whether the part is "good" or "bad." Then when you investigate the different vision algorithms, you'll be able to see which vision systems can perform the quality inspection tasks you need.

Of course, there are alternatives to using vision for quality checks. We've listed some options below, along with how they compare to the use of a vision system:

- **CMM or Coordinate Measuring Machine**—CMMs are the most commonly used systems for tasks like validating the dimensions of a machined part. In that case, the system touches different surfaces on the part to measure it.

  People usually look for CMM machines with an accuracy of approximately 0.25 mm. It's pretty difficult to reach this level of accuracy with a vision system – and for the moment, it's probably quite expensive.

  You can see why it's so expensive when you consider the following. Let's assume you need at least 3 pixels to attain an accuracy value of 0.25 mm, and that we have a 150 x 150 mm field of view, which is large enough to cover the part. You would need an 1800 x 1800 pixel camera to do the job. This doesn't even take into account any lighting challenges that may be involved with the shiny metal objects coming out of a CNC machine. For these reasons, if you need to validate part dimensions with high precision, CMMs are probably the way to go.

- **Eddy Current**—This technique can be used to detect a part's position and its defects. Thanks to its high sensitivity, you'll be able to detect small cracks, deformities, pits, etc. However, eddy currents are limited to conductive materials, whereas vision systems are not.

- **Vision**—If you're on a production line where products are coming out on a conveyor and you simply need to check for part presence, then vision is the best option. That's because the camera can take a snapshot of parts while they're moving on the conveyor, so you can perform live quality checks without having to remove the parts from the production process. Instead of doing batch sample testing, you're able to test every product coming down that line. Vision will also be useful for performing measurement analysis (without extreme precision), or checking for a feature's presence.

To help you judge whether vision is a good fit for your process, here's a list of vision functions that are often used in quality inspections:

- **Pattern Matching**—With this function, you teach a few "good" parts and a few "bad" parts to the camera. The camera sets a pass/fail threshold. Then, when it takes an image of a new part, it indicates if this part meets the threshold. This function is often used when you need to check a part for deformities or for the presence of a specific object (e.g. holes in a part, or a cap on a bottle).

- **Color Matching**—Some vision systems, like the Robotiq Wrist Camera, can check the color of an object. Color matching improves model detection and enables object sorting during the part's manipulation, helping you make the most of each robot cycle.

- **Optical character recognition (OCR)**—This is often used to confirm that what was printed on the object is visible and written correctly.

- **Measuring**—Vision systems can include various ways of performing measurement checks through the use of images. These methods include measuring the distance between edges, the distance between holes, etc.

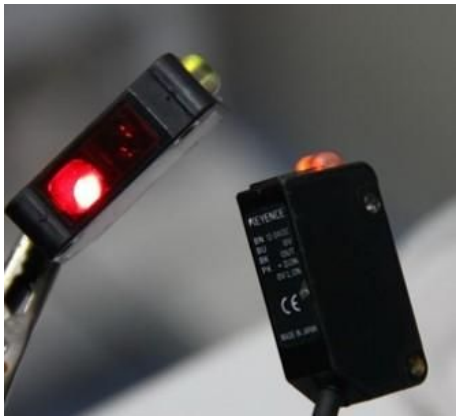
- **Filters**—Filtering techniques like blob analysis enable the system to detect visual defects (like pits, holes, and cracks). For more on this, refer to the section on Image processing terms.

- **Counting**—This is sort of like measuring, but instead of measuring the distance between edges or holes, it involves counting how many objects there are, or how many holes are present on a part, etc.

What Kinds of Vision Systems Are Available?

2D cameras aren't the only vision systems out there. This section covers all the options, from 1D to 3D.

1D Technologies



A 1D sensor is sufficient for tasks like measuring the height of a part that's moving on a conveyor. In this case, one type of sensor you could use is a single laser point to measure the distance from the sensor to the part's surface. A single laser point will point on the surface and (in general) use triangulation to obtain the distance between the part's surface and the sensor. With this kind of technology, it's only possible to get data for one axis.

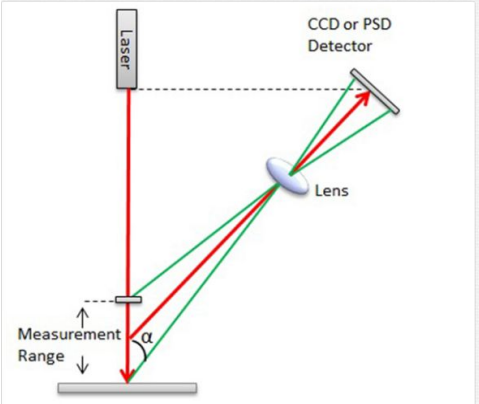


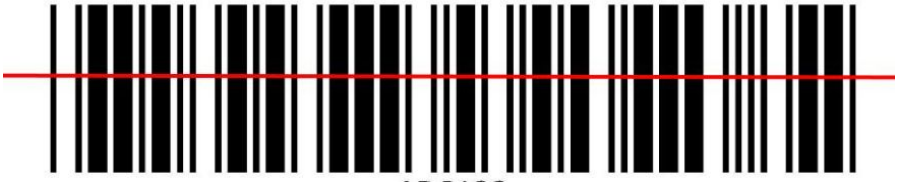
FIGURE 1.
Principle of Triangulation.

Another type of 1D sensor is a line-scan camera. It can also be used for inspections, as this article notes:

> *"If you need to inspect the surface of a part that is moving on the conveyor, a line-scan camera would be preferable: you would install it on top of the conveyor. You would gather "line images" and stitch them together in order to make a 2D image, and analyze this one. The linear camera is similar to a 2D camera, but with a sensor that has a single row of light-sensitive pixels."*

Keep in mind that for the system to stitch together the various scans to get the resulting image, it usually requires some sort of input to figure out the speed of the conveyor. Most of the time, this kind of system will take the pulses from the conveyor motor to figure out the distance between the scans. Some systems have built-in features that allow you to connect the conveyor to the sensor.

A third type of 1D sensor, called a barcode reader, throws a laser line at the image or code and reads only the data along this line. The sole reason for the height of a UPC code (which is one type of barcode) is to allow a certain range of flexibility for the scanner, so that it can tolerate a scan line that's not perfectly horizontal.

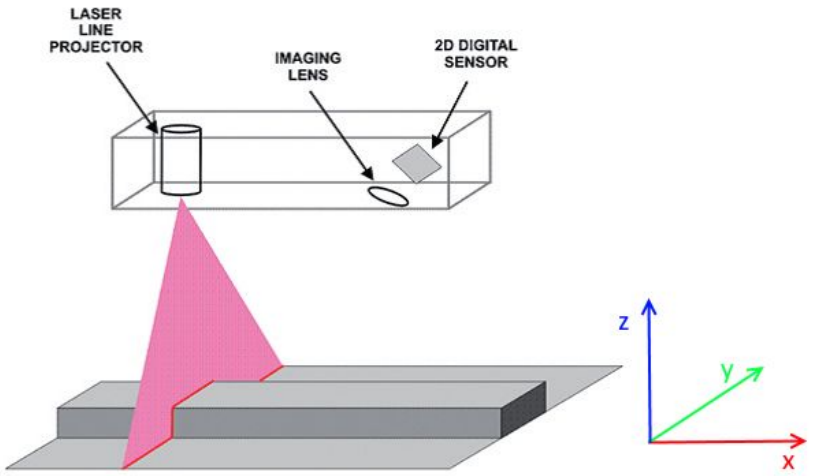


2D Technologies

2D scanners represent the middle of the price range. They're similar to the cameras we use in everyday life, in that they are essentially rectangular sensors that give rectangular images. This eBook focuses on this kind of vision sensor. Here's a list of various 2D sensors:

- **2D Cameras**—These provide the X and Y coordinates for each pixel value in the images they produce, which is why they're considered 2D systems. There are many choices on the market regarding various resolutions, grayscale vs. color, completely integrated vs. independent cameras, etc. Find out more in the next section.

- **2D Laser Scanners**—This technology is similar to the 1D single point laser described in the 1D technology section. However, instead of having a single laser point, here a laser line is analyzed. So you end up with distance measurements from the sensor (Z axis) for the whole width of your object (Y axis). Such 2D scanners are sometimes sold as "3D profilers," because they can be combined with the movement of the part on a conveyor to determine the other axis (X axis).



The principle is the same as for the line-scan camera we saw above: hook up the encoder pulses from the conveyor to the vision system, and stitch the scans together using this speed information.

The number of dimensions you can measure (1D, 2D, or 3D) can be a bit confusing, since you can also find 4D scanners on the market. These take multiple 3D measurements (3D + the time variable = 4D).

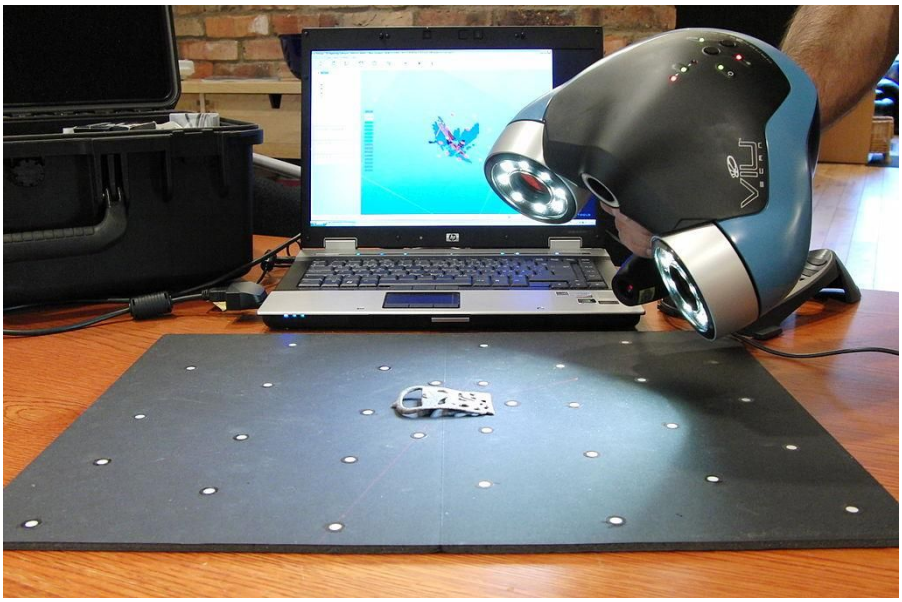
If you want to know the real number of dimensions, focus on the sensor itself. In this case, you want laser distance sensors that measure the distance (first dimension) along a line (second dimension) – which means you want a 2D scanner.

3D Technologies

Another option is the 3D scanner. These scanners are used for reverse engineering, metrology, exact measurements (e.g. of defect depth), etc.

3D scanners use various techniques, including Shape from Shading (the process of computing a three- dimensional shape of a surface from one image of that surface), laser triangulation, and structured lighting. You can read more about them here.

Depending on the technique, you can either get approximate data within a few seconds, or more precise data in several minutes. Here are two widely used techniques:



- **Structured Light Sensors**—This type of sensor projects some sort of structured light pattern onto the part. The principle is quite simple: deformations in the line of light represent bumps or dents in the object. 3D data are obtained via an analysis of variations in the light pattern. Usually, two 2D cameras are used to analyze the structured light pattern. Then algorithms combine the data to create output information. Here is a video showing this kind of technology in action.

- **Time of Flight Sensor**—This technology involves shining light on the part to be measured and then measuring the time it takes for the light to travel to the object and back to the camera's sensor. 3D points can be obtained from the "time stamps" for various points on the part. These systems are usually less accurate than structured light sensors. Here is a video showing how this process works.

Choosing the Right 2D Vision System for Your Application

There are two main categories of 2D sensors you can buy: intelligent cameras (also called smart cameras) that have the sensor and processor embedded in one casing; and stand-alone or independent cameras that only contain the sensor, and for which the processing needs to be done on another device.

Wondering why anyone would buy the latter? Well, they both have their advantages and drawbacks, so it really depends on your needs:

- **User's level of expertise in machine vision and programming**—Smart cameras tend to be easier to program than individual camera systems. They have built-in vision tools that are easy to configure for your application. Some systems use a simple web-based interface where you connect to the camera using the camera's IP setting. These systems can output things like pass/fail status, object localization, etc.

  Non-integrated cameras will require additional software to do the image analysis. Their advantage lies in providing greater flexibility.

- **Image processing**—Non-integrated cameras will require an external processor to analyze the acquired images. You could plug the camera into a computer to process the images. This should be a relatively quick process because computers generally have more processing power than smart cameras. But that doesn't mean computers are always best: for simple vision analysis, like barcode reading, smart cameras probably have enough processing power. This is one reason why you should determine what processing speed you need first before buying a vision system.

  Another image processing factor to consider is flexibility. An embedded software library in a smart camera will provide you with basic image processing functions. Depending on the camera, you will have access to some filters, pattern recognition, teaching of a template image, etc. For each of these functions, you will be able to configure the parameters, such as region of interest (region of the image you want to process), acceptable thresholds, etc.

  If you need more sophistication in your image processing – such as the ability to design your own filters – you will need to have access to a more flexible and advanced library. A non-integrated camera with a separate software library will offer more options, as you will have access to more image processing functions and parameters. Moreover, you'll have the flexibility to code your own image processing algorithm, as you will have access to the complete data.

- **Price**—Do you think having high-resolution images with a full-blown processor sounds cool? Of course it does, but you'll spend a lot more money on this kind of setup, plus you'll need to synchronize your processor with your camera. Maybe think twice before spending money on a setup that might be overkill.

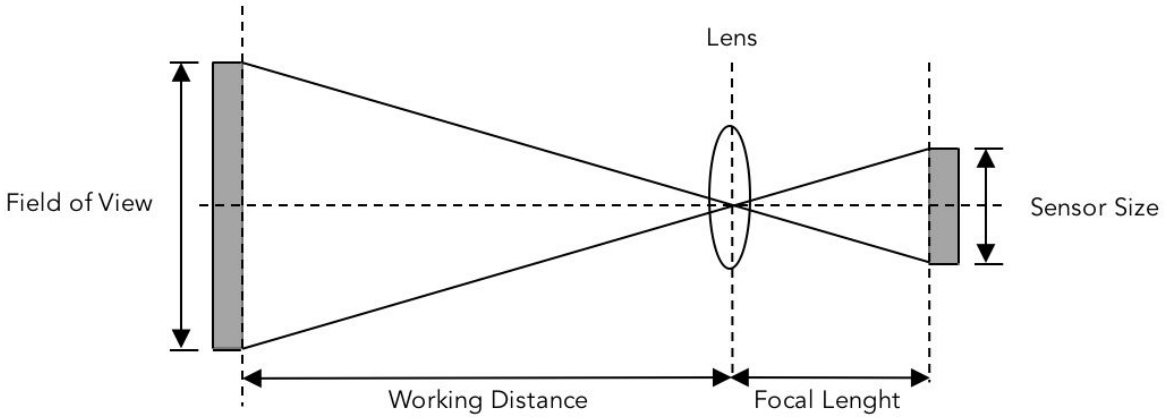
  When comparing the costs of one solution to another, you need to take into account the integration time, the software setup time, and the programming time. Then you'll be able to determine if a smart camera or a more adjustable independent camera is best for you.

  Now you might wonder: where does the Robotiq Wrist Camera stand in relation to the descriptions above? It is a completely integrated camera, like a smart camera. It's been

specially designed for ease of use. Check out its specifications and get more information here.

Image Acquisition Terms

When comparing various cameras, or when configuring one, you might come across some of these image acquisition terms. Here are some basic definitions so you can understand what exactly you're comparing!



Resolution: The number of pixels in an image. The Robotiq camera has a resolution of 5 megapixels (5 mpx), because the sensor has 2592 x 1944 pixels = 5,038,848 pixels = approx. 5 mpx.

However, as noted on Wikipedia, the term "resolution" can sometimes be confusing:

> *"Unfortunately, the count of pixels isn't a real measure of the resolution of digital camera images, because color image sensors are typically set up to alternate color filter types over the light-sensitive individual pixel sensors. Digital images ultimately require a red, green, and blue value for each pixel to be displayed or printed, but one individual pixel in the image sensor will only supply one of those three pieces of information."*

Despite this, most camera manufacturers still use "resolution" to describe the number of pixels.

Camera sensor size: The physical size of the sensor. Digital cameras often have smaller sensors than the classic 35 mm sensors. The typical sensor size range in digital cameras goes from 5.8 x 4.3 mm to the full 36 x 24 mm (called the "35 mm sensor" – yep, even though it's 36 x 24 mm, not 35 mm, the name stuck).

For an equivalent number of pixels, a bigger sensor will have bigger pixels. The bigger the pixels, the more light the sensor will receive. However, for the same size of sensor, having more pixels also means having a bigger resolution. As we've mentioned a few times, it's all a matter of balance and what you want to achieve!

Focus: The object plane's distance from the sensor where the sharpness of the image is optimal. Focus is probably easiest to define in reference to its opposite: a blurry image is an image that's out of focus. (Well, technically blurriness could also be caused by a fast-moving object and a long exposure time... but we'll get to that later.)

Field of view (FOV): The width and height of the object plane.

Depth of field (DOF): The distance range, on the object plane, where the focus is at an acceptable level for the object to be seen.

Working distance: The distance from the object plane to the focus point. Note that the closer your object plane is to the camera (the smaller the working distance), the shallower the depth of field you'll have.

Contrast: The difference in brightness between the object of interest and the background. The more contrast you have in your image, the easier it will be to distinguish your object of interest. Several lighting techniques will provide higher contrast. Among these, a widely used technique is selecting and positioning a lighting source that will reflect on your object of interest and in turn reflect light back to the camera ("direct lighting"). For best results, use a dark background.

Another method is the backlight technique: you position the object of interest between the lighting source and the camera, which enables you to see the object's contour as a shadow, while the background is almost pure white. (See Where to Start for an example.)

Aperture: The physical aperture of the lens determines how much light is let inside the camera. The aperture influences the depth of field (DOF): the lower the aperture, the deeper the depth of field you'll have.

Shutter speed, also called exposure time: The amount of time the lens stays open. If you want to take a snapshot of a moving object, you will need to keep the exposure time to a minimum, or else it will be blurry.

Exposure: This is defined as "the amount, and act, of light falling on photosensitive material" – so this is a combination of the effect of ambient light (both natural light and artificially-added light), aperture, and shutter speed.

Frame rate: The maximal number of frames, which is generally expressed in FPS (frames per second). When comparing frame rates, make sure you consider the "resolution" parameter as well, because the frame rate will be lower when images are captured at full resolution.

RGB (red-green-blue): The three primary colors that are combined to create a broad array of colors. RGB cameras capture the color information, whereas grayscale cameras do not.

Trigger: The automatic release of the shutter. Triggers are really helpful when you want to synchronize the capturing of the image, e.g. to take a snapshot of an object moving on a conveyor belt.

They're also good when you need to use an external light. It's often challenging to get the maximum amount of light from a flash, in a really condensed amount of time. You don't have a big window of time to snap the correct picture: the flash will be available at a specific time, for, say, 3 ms, and the moment when the picture is taken must be perfectly synchronized with it.

Using a trigger means the shutter speed can be set fast, which will have the effect of "freezing" the movement of the object in time.

Well, that was a lot to cover! If you need help selecting or configuring your vision system, start with this article or contact our applications engineers.

Image Processing Terms

Your brain goes through a lot of processing in order to get from "seeing something" to "recognizing what it is." A big part of image processing is accomplishing something similar using various functions.

This section covers some image processing terms that will help you keep up during a conversation with a vision expert (should you happen to run into one). Note that we've used grayscale images here – however, the same principles apply to color images.

- **Thresholding:** You input an image and a threshold value ranging from 0 (black) to 255 (white). Let's say our threshold is 150. In the output image, dark pixels that were assigned a value between 0 and 150 will be set to 0 (black), and lighter pixels that had a value between 151 and 255 will be set to 255 (white). Thresholds are often used after the image has been filtered.

- **Filtering:** Filtering is often used to reduce image noise. It can also be used to dampen non-useful details in the image. Filtering often uses neighboring pixels to compute the output value of a single pixel. For example, an averaging filter might use the value of eight neighboring pixels. It computes their average, which becomes the value of the pixel in the new output image.

  In our example below, the center pixel, which has a value of 62, will be replaced with the value of 55 in the new output image (because 55 is the average of all the neighboring

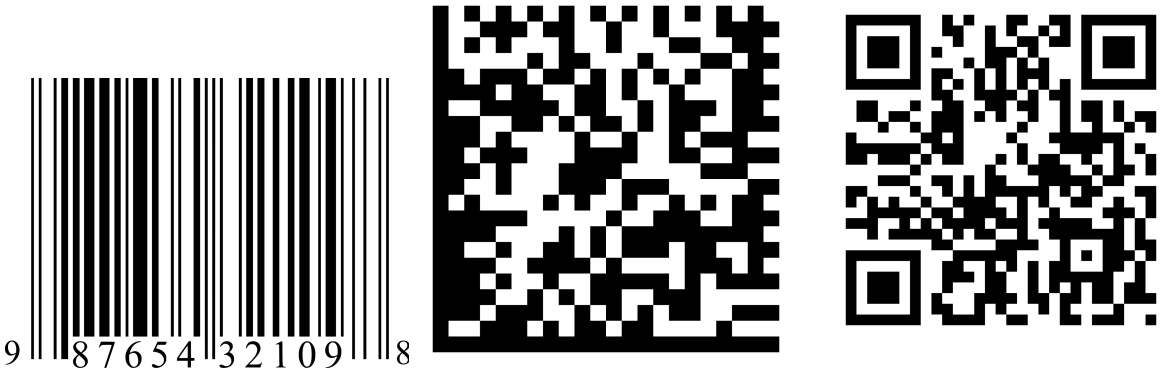
pixels). In this way, neighboring pixels will be more similar to one another, which will generate smoother contours.

48	60	55
40	62	60
60	54	59

- **Detecting contours:** In this function, the algorithm searches for a line or a circle, etc., depending on what you have programmed it for. For instance, it could look for a vertical line by reading the image from left to right and searching for a light-to-dark contrast. The program will do this on many horizontal lines, spaced out at a predetermined interval (e.g., every 50 pixels). The output should be the detected line (although it might be a bit uneven, since the identified points might not be perfectly aligned). Contour detection can be even more complex: check out this link for more.

- **Histogram analysis:** An image histogram is the graphic representation of the frequency of each pixel's value. The X axis represents the grayscale value, from 0 to 255, whereas the Y axis represents the number of pixels that have a given value (from 0 to 255). You could use a histogram to decide on a thresholding value, or you could use it to determine whether the color of the part corresponds to what is expected.

- **Barcode reading:** Various information can be encoded in a barcode, like the date of manufacture, lot number, etc. We are used to seeing UPC barcodes on items we buy. But 2D barcodes, like Data Matrix or QR codes, are also used frequently because they can hold much more information. Barcode reading is the process of scanning the code and decoding its information. Its output is usually a series of letters and numbers.



Left to right: UPC code, Data Matrix, and QR Code.

- **Template matching:** An image processing algorithm is used to search for a match with a known shape. First, you will need to "record" the shape you want the program to recognize by taking a few template images. Then, the "part model generator" will extract features from these images and create a model with this information in the software library, so it can recognize this model later on.

  Using multiple images will allow the part model generator to distinguish between a "feature" and a "shadow." When the camera/image processor is programmed to use template matching, it searches in its image bank for the model that was previously recorded. Once it finds a match, it flags the image. This information can then be used to count the number of objects or compute the object's location, or to generate a robot path for part picking. You can read more on template matching at the following links: a post on our blog, a document by Adaptive Vision, a research paper using LEGO bricks, and a PhD thesis.

- **Optical character recognition (OCR):** This means recognizing letters and numbers based on a library of templates. The recognition will be sensitive to such things as the font being used.

- **Blob analysis:** This is a method of analyzing connected pixels. Blob analysis is frequently used with other tools such as filtering and thresholding. By using the black-and-white image that comes out of the thresholding algorithm, the computer should be able to easily analyze the shape of the blobs. Find out more here.

- **Region of interest (ROI):** This refers to cropping an image in order to analyze a portion of it. The smaller the region to analyze, the faster you'll be able to process your image. If your field of view is fixed, and it's too big for your needs, it's a good idea to select a smaller region of interest.

Template Matching Challenges

Now that you know how template matching works, it will be easier for you to understand what constitutes a challenge for this kind of algorithm. Some of these challenges are discussed below.

Overlapping Parts

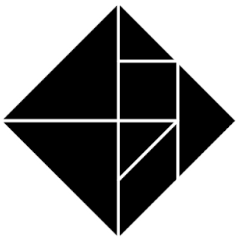
Of course, non-overlapping parts are easier to recognize than overlapping ones. This doesn't mean it's impossible to detect overlapping parts, but is more of a challenge for the algorithm.

With overlapping parts, you might end up with multiple possibilities during contour detection. So in terms of reliability, you have a much better chance of recognizing non-overlapping parts every single time.

To understand this more clearly, let's bring back a childhood game: tangram puzzles. I'll give you an image of a tangram puzzle, and you have to guess which parts were used to assemble it (and how they were put together).

Here we go:

This letter E has been put together from an unknown number of the following pieces:



So which of these parts were used, and where are they located?

Here's the answer:

Admittedly, the capital E example is hardly realistic. None of the parts overlap, they fit together perfectly, and they're all easy to distinguish.

Still, it illustrates how hard it can be to recognize shapes – even though you have a "library" of known shapes and a picture of the current shape you want to recognize. Plus, although the parts were just touching each other, not overlapping, the contours were still hard to determine.

We haven't even brought up the fact that if there's no contrast between two parts, then the camera will have a hard time determining which one is on top of the other, so the robot won't know which one to pick up first!

Insufficient Contrast

If you want to detect edges easily, or recognize variations in grayscale, you'll obviously need strong contrast. Therefore, the background for the parts should be as different as possible from the parts themselves (in grayscale).

Want to detect black parts? Use a white background.

The parts' surface finish will also influence the contrast. If you have relatively dark grey metallic parts, you might opt for a white background. However, if the parts are shiny, there's a good chance they'll reflect light back into the camera, especially if they're caught at a certain angle.

So, depending on the camera angle, you might end up with a part that appears almost white (from the light reflected back to the camera), relatively dark, or a mix of the two. This is another type of challenge for template matching.

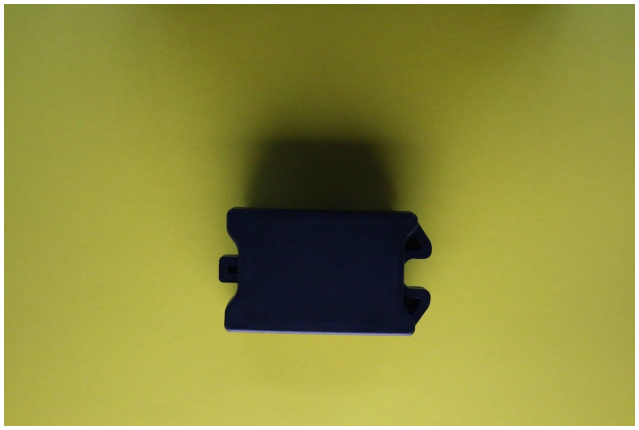
External Light Sources

If your part's material is highly reflective, you should also consider lighting that comes from outside your vision system, like sunshine coming in from a window, or flashes of light from a nearby soldering process.

Even if your part's material is not that reflective, ambient light changes could affect the shadows in the background. This can cause problems when shadows in the background are mistakenly identified as being part of the object you want to detect.

So make sure you take external lighting into account, and read up on these tricks to cope with such problems.

Shadows



Shadows create another visual challenge! They can still be managed, but they might reduce the robustness of the part recognition program.

Make sure your template image has as few shadows as possible. Otherwise, your system might end up looking for a part *and* its shadow, which will be difficult to find, since the shadow is unlikely to appear identical every time.

One way to compensate for shadows while teaching is to use multiple images of the same object taken at various angles. For example, you could take one image, rotate the part, take another image, and so on. Then, build the "template image" from a combination of all of these images.

Too Much Variation Between Parts

This is a crucial aspect of template creation, and it harkens back to our blog post on optical character recognition (OCR):

> "Let's look at an OCR example: you have taught the system that the pattern to be looked for is 8, but a B would also be okay because you know the left-hand side of the character sometimes has problems being punched correctly.
>
> Now let's say the machine reads a 3... would that be acceptable? The left-hand side is different, but you have trained the system to be less picky on this side, because of known punching problems... That's a good example of desensitization: the more various things you input into the system as 'normal,' the less sensitive your system will become."

The example here is about recognizing a specific letter, but the same rules apply to other shapes. If you are too permissive at the teaching stage, the "template" will be less specific and the part recognition results might be less reliable. If you have similar part models, it's likely that the system will confuse a few of them.

Another example is deformable objects. Think of how a bath towel could be presented: folded, or a shapeless blob of cloth. In the latter case, there are so many possible towel configurations that the system will probably end up identifying anything as a towel.

Where to Start

We definitely recommend starting slowly, so you can gain confidence from your first experience.

Here are some tips for easing into your first vision project:

- Start with a simple pick and place application.
- Use parts that have:
    - Few or no variations between parts from the same model, but great variations from one model to another.
    - A distinct contour.
    - Recognizable features (e.g., holes drilled at specific and consistent spots).
    - A matte (non-shiny) surface.
- Make sure there's a strong contrast between the part and whatever's behind it.
- Take control of the lighting. Don't place your system near a window or a ceiling light that could be turned on and off during the day. Test your system and see how it behaves.

- Place parts so that they a gap between them (no overlapping, no touching).
- Pay special attention to the teaching phase. Treat different surface finishes as different shapes to be taught.

If you start teaching your vision system with an easy challenge, you can get used to its parameters and experiment with the kinds of variations your system can tolerate. For example, you might find that overlapping parts are okay in your application, if the parts have other features that are easy for the system to recognize.

Here are some ideas for once you're ready to take it to the next level:

- If surrounding light or undesirable reflections are a problem, you could:
    - "Hide" your system beneath a hood of some kind, or, if your system is enclosed in a robotic cell, use an opaque top and tinted windows. There's no point going to all this trouble if it's not needed though, so test first and judge afterwards!
    - Add another light source. You will need to control this equipment and synchronize it with the timing of the camera. If, for example, you work with highly reflective metallic parts, and are only interested in detecting their contours in order to pick them up, you might find it easier to backlight the parts instead.

The principle of backlighting in photography: the light source is placed behind the subject of the photograph, so the subject appears black on a light background with an easily detectable contour.

- Upgrade from a simple pick and place project to a "check and pick and classify" process by verifying certain features. For example, the process could be to check that a label has been applied to the object, then put it in tray A if it has a label, or in tray B if it doesn't.
- Use your vision system to count features or parts.

- Check for a specific shape, then flag any other shapes that might represent defects. Play with tolerances to make the proper adjustments to your system.

Tips and Tricks for Using the Robotiq Wrist Camera

Robotiq's camera has pre-configured settings that let you focus your energy on your application.

Even though you'll probably be able to accomplish most of what you'd like to do with vision, you might encounter some applications where you'll need to play with the system's limits.

Here are some examples of "pushing the limits":

1) Your object is larger than the camera's field of view.

In many situations, it is possible to locate an object that's bigger than the field of view by using a workaround. One method is to teach a distinctive feature on the object that fits in the FOV, such as a written word or logo that's always located in the same place, or a specific shape that's at a constant distance from the object's contour.

Of course, this will only work for tasks that don't require a complete view of the object (like locating the object or checking for its presence). A workaround for top-to-bottom inspection tasks is to take many snapshots in a grid pattern to form a complete view of the part.

2) Your object is very shiny.

There's a simple solution: change your part's appearance by painting or powdering it to reduce shine. Unfortunately, this is often incredibly inconvenient.

Fortunately, as long as the reflections aren't hiding any significant edges, you can simply erase the mask over the reflective area in the Edit Model view of URCap, and those reflections will be ignored.

Another possible solution is to diffuse the light (such as with a photography umbrella) or change the light source. The goal here is to reduce the amount of light being shone being reflected off your part. Another strategy is backlighting (see Where to Start for details).

Robotiq's first Wrist Camera had a fixed light setting, which caused problems with reflective parts. The new Wrist Camera lets you turn off the flash, so now it's easier to manage highly reflective objects.

Another option is to relocate your vision operation within the factory (to a place where you'll have more control over the lighting), or move it to a different stage in the manufacturing process.

For example, if you want to check the presence of a part feature and have placed this operation after the polishing stage, you could rearrange things so the checking operation takes place *before* the polishing stage, when the parts still have a matte appearance.

3) Your application requires specific lighting or focus settings.

The original Wrist Camera had an automatic mode that chose most of the lighting and focus calibration settings for you. But since some applications require more flexibility, we've added a manual focus option to the new camera. Focus can be adjusted for the Teach Model phase and then optimized to another fixed setting for runtime conditions or switched to automatic focus.

Also, we've refined the automatic mode for object recognition. Now you can highlight areas you want to keep in your object definition, or erase details you want to ignore, all through the Teach Pendant. You can also choose to use the area that was automatically selected or select it manually, depending on what's best.

4) You have changing backgrounds that make it hard to detect objects.

The new URCap version of the Wrist Camera includes the option to use color validation in addition to edge detection. This makes detecting and locating objects more reliable in changing background conditions, and greatly reduces the rate of false positive detections.

You can also use color validation to get your robot to perform more tasks for you. With color validation, a single picture can tell is enough to tell whether the shape and color of a part meet expectations. For instance, you can differentiate anodized parts from bare metal ones and sort accordingly, or do color-based kitting of identically shaped parts – all in a single operation.

Conclusion

Hopefully you now understand a bit more about machine vision and how to choose the correct device for your application. Want to learn more? Check out Robotiq's blog for other insightful articles on machine vision and robotic applications.

About Robotiq

Robotiq's Lean Robotics methodology and products enable manufacturers to deploy productive robotic cells across their factory.

They leverage the Lean Robotics methodology for faster time to production and increased productivity from their robots. Production engineers standardize on Robotiq's Plug + Play Components for their ease of programming, built-in integration, and adaptability to many processes. They rely on Flow's software suite to accelerate robot projects and optimize robot performance once in production.

Robotiq is the humans behind the robots: an employee-owned business with a passionate team and an international partner network.

Let's Keep in Touch

For any questions concerning robotic and automated handling or if you want to learn more about the advantages of using flexible electric handling tools, contact us.

Join us on social media:

Workfloor: Robotiq's Blog

Twitter

Linkedin

Facebook

Youtube

Google+

DoF

Robotiq's community where industrial **automation Pros** share their **know-how** and **get answers**

Ask Your Question

LEARN MORE

robotiq.com | leanrobotics.org