3

How a Robot Works

This chapter is written for readers who have limited technical background about intelligent interactive robotics. More specifically, what is covered in this chapter:

- the basic hardware and software components that a robot consists of;
- the techniques we can apply to make a robot ready for interacting with people.

As a way of thinking about how a robot works, let us role play by imagining being a robot. We might think we can do a lot of things, but we soon find out our capabilities are severely limited. If we are newly built robot, without appropriate software, our brains are completely empty. We cannot do anything—move, know where we are, understand what is around us, even ask for help. We find the experience of being a robot rather strange and difficult to imagine. The main source of strangeness is that the new robot's brain is nothing like a human brain, not even an infant's. The robot has no basic instincts, no goals, no memory, no needs, no learning capabilities, and no ability to sense or act. To make a robot system, we need to integrate, and at least partially develop, hardware and software together to enable the robot to sense and act in the world.

In this chapter, we look at the common components of a robot and how they are connected to enable participation in interaction. Section 3.1 explains basic ideas about the components needed to build a robot. Section 3.2 explains the types of hardware. Section 3.3 introduces sensors, such as cameras, range finders, and microphones, and Section 3.4 introduces actuators. Finally, Section 3.5 explains the software that accompanies the hardware elements, which addresses the perception (e.g., computer vision), planning, and action control of the robot.

This material has been published by Cambridge University Press as Human Robot Interaction by
Christoph Bartneck, Tony Belpaeime, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Sabanovic.
ISBN: 9781108735407 (http://www.cambridge.org/9781108735407).

3.1 The making of a robot

To build a robot, one of the first steps is to establish connections between the robot's sensors, computer, and motors so that the robot is able to sense, interpret what it senses, plan actions, and then act them out. Once the robot is connected, to a camera, for example, its computer can read the data the camera provides. But the camera image is nothing more than a large table of numbers, similar to the following table:

9	15	10
89	76	81
25	34	29

From these numbers, can you guess what the robot is seeing? Perhaps a ball, an apple, or a fork? Assuming that each value in the table represents the lightness value of one sensor element in the camera, we can translate those numbers to a graphic that is more meaningful to humans (see Figure 3.1), but the graphic remains meaningless to the robot.

You might be able to see a line in the image shown in Figure 3.1, but a robot has no understanding of what a line is. This line might be the edge of a cliff from which the robot could fall and damage itself. But the robot does not have a concept of height or gravity. It would not comprehend that it could fall if it crossed this line. It does not know that if it fell, it would likely come to rest upside down. It would not even recognize that its arm would be broken. In other words, even concepts that are vitally important for interacting with and surviving in the world around us that are innate in humans have to be explicitly programmed in a robot.

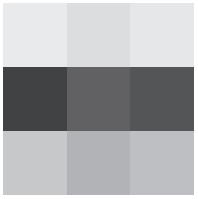

Figure 3.1 The camera's data translated into a grid of grayscale pixels.

A robot, in essence, is a computer with a body. Any functionality needs to be programmed into the robot. A problem that all robots have to deal with is that although their sensors and motors are sufficient for operating in this world, their intelligence is not. Any concept of interest to roboticists needs to be internalized, that is, programmed into the robot. This requires a lot of time and effort and often involves many cycles of trial and error. The analogue world out there is converted into a digital world, and translating tables of numbers into meaningful information and meaningful responses is one of the core goals of artificial intelligence. Being able to identify a face from a large table of values, recognizing if a person has been seen before, and knowing that person's name are all skills that require programming or learning. Thus, the progress of human–robot interaction (HRI) is constrained by the progress that is made in the field of artificial intelligence. Robotics engineers integrate sensors, software, and actuators to enable the robot to make sense of and interact with its physical and social envi-

This material has been published by Cambridge University Press as Human Robot Interaction by
Christoph Bartneck, Tony Belpaeime, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Sabanovic.
ISBN: 9781108735407 (http://www.cambridge.org/9781108735407).

ronment. An engineer might, for example, use accelerometer sensors, which can detect acceleration and the Earth's gravitational pull, to read the orientation of the robot and determine if it has fallen. A cliff sensor, consisting of a small infrared light source pointing down and a light sensor, can be used by the robot to avoid falling down a staircase.

Typical problems that robot engineers have to solve for the robot include the following:

- What kind of body does the robot have? Does it have wheels? Does it have arms?
- How will the robot know its location in space?
- How does the robot control and position its body parts—for example, arms, legs, wheels?
- What does the space around the robot look like? Are there obstacles, cliffs, doors? What does the robot need to be able to perceive about this environment to move safely?
- What are the robot's goals? How does it know when it has achieved them?
- Are there people around? If so, where are they, and who are they? How will the robot know?
- Is a person looking at the robot? Is someone talking to it? If so, what does the robot understand from these cues?
- What is the human trying to do? What does the person want the robot to do? How can we make sure the robot understands this?
- What should the robot do, and how should the robot react?

To address these questions, HRI researchers need to build or choose appropriate hardware and an appropriate morphology for the robot, and then develop relevant programs—the software—that can tell the robot what to do with its body.

3.2 Robot hardware

At the time of this writing, a number of robots have been produced for the consumer market. Although not all of them may have become domestic staples, these commercial robots are often suitable platforms for HRI research. Commercially available robots provide a variety of body types, including animal-like, humanoid, and more mechanical.

Aibo, an example of an animal-like robot, looks like a dog with a somewhat mechanical appearance (see Figure 3.2) and has the ability see, hear, feel touch, make sounds, wag its ears and tail, and move around on its four legs. The first Aibo models were sold in 1999, and sales were discontinued in 2006. Eleven years later, sales of new models started again.

Pepper, on the other hand, is an adolescent-size humanoid (see Figure

This material has been published by Cambridge University Press as Human Robot Interaction by
Christoph Bartneck, Tony Belpaeime, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Sabanovic.
ISBN: 9781108735407 (http://www.cambridge.org/9781108735407).

Figure 3.2 Aibo ERS-1000 robot (2018–present). (Source: Sony)

3.3). Some stores use Pepper to attract visitors and market wares and services. The company that produces Pepper also has the smaller Nao humanoid (see Figure 2.4) available for consumer purchase.

A more mechanical-looking robot, the K5 security guard robot is commercially available in the United States and is one of the few robots that are meant to be used outdoors.

Robots that were not explicitly designed to be used for HRI can nevertheless still be used or even modified for HRI studies. The most commercially successful home robot is still the iRobot Roomba vacuum-cleaning robot, millions of which have been sold around the world. Roombas not only are an interesting agent for use in studying the public's relationship with robots (Forlizzi and DiSalvo, 2006) but have also been modified and hacked for HRI research. iRobot also makes a programmable version of the Roomba, the Create, which lacks the vacuuming component and is used in research and educational applications of robots.

Telepresence robots can also be used as platforms for HRI research. Many different types exist on the market, including mobile versions such as the Beam and desktop versions like Kubi. Small mobile robots carrying a screen displaying a friendly face are being developed, soon to be ready for release in the consumer market.

Although commercially available robot hardware provides a wide variety of morphologies and sensing and programming capabilities, every robot is limited in what it can do; its appearance and capabilities constrain the interactions it can engage in. Researchers, therefore, also conceive and build their own robots, which range from simple desktop and mobile platforms with or without a manipulator to very humanlike android robots. The choice of a particular morphology for a robot to be used in HRI research often depends on the capabilities needed for the expected task (e.g., whether it needs to be able to pick up objects), the type of interaction (e.g., petlike interactions can benefit from an animal-like robot), and people's expectations and perceptions of differ-

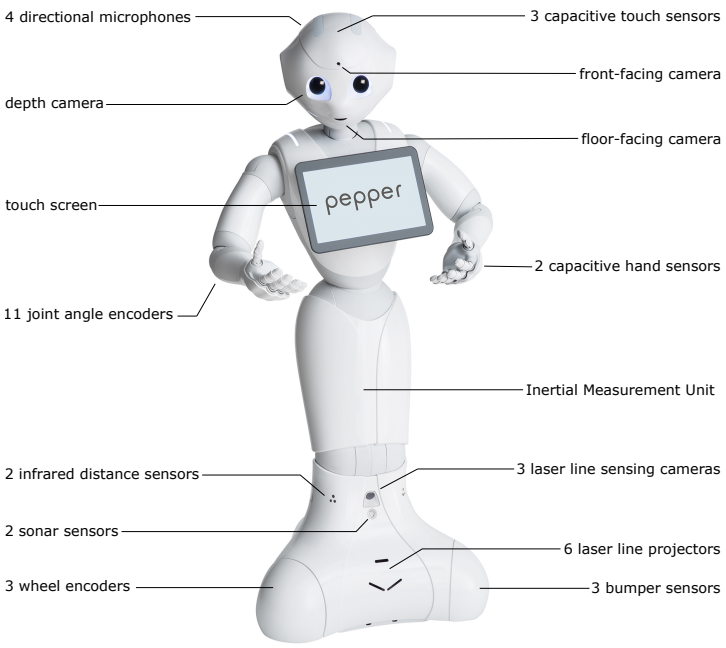

Figure 3.3
Pepper robot (2014–present) and its sensors (Source: Softbank Robotics and Philippe Dureuiltoma)

ent morphologies (e.g., humanoids may be expected to behave and be intelligent in ways similar to humans).

3.3 Sensors

Most social robots are equipped with sensors that allow them to gauge what is happening in their environment. Many commonly used sensors are related to the three most commonly used modalities in human interaction—vision, audition, and touch—but robots are not at all limited to human modes of sensing. It is often helpful, therefore, to consider what types of information the robot needs to perceive and what the most accurate and expedient ways are for it to do so, rather than focusing on reproducing human capabilities.

3.3.1 Vision

Camera

A camera consists of lenses that focus an image onto a sensor surface. The sensor surface is implemented using either a charge-coupled device (CCD) or, more often, a complementary metal-oxide-semiconductor (CMOS) technology. The basic element of a camera is a light sensor consisting mainly of silicon that converts light into electrical energy.

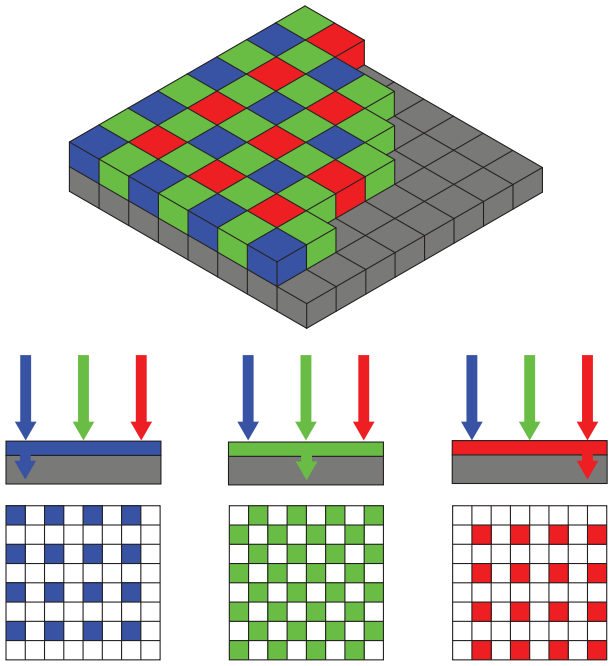

Figure 3.4 Array of CCDs in RGB camera.

A camera consists of an array of millions of these light sensors. Typically, color in a camera image is represented using three values, red (R), green (G), and blue (B). Hence, a camera is commonly referred to as an RGB camera. The sensors on the sensor surface are not sensitive to the color of the light hitting them; they are only sensitive to light intensity. To make an RGB camera, small color filters are placed on top of the sensor surface, with each filter letting through only red, green, or blue light (see Figure 3.4). Cameras are the richest and most complex sensors available to robots, and through its wide adoption in digital cameras and smartphones, the RGB camera has become miniaturized and very cheap.

In computer vision research, investigators often put cameras in the environment to facilitate accurate vision. Although this is one of the realistic approaches to yielding stable performance from computer vision, in the HRI setting, it is sometimes discouraged because people can feel uncomfortable around cameras. For example, in a project in which elderly people were being assisted in their home by a robot, the engineers would have loved to have cameras on the robot and in the home because it would have allowed the robot to accurately track and interact with people. However, the elderly participants were quite firm in their refusal of the installation and use of cameras, forcing the team to use localization beacons and laser range finders instead (Cavallo et al., 2014).

This material has been published by Cambridge University Press as Human Robot Interaction by
Christoph Bartneck, Tony Belpaeime, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Sabanovic.
ISBN: 9781108735407 (http://www.cambridge.org/9781108735407).

Most cameras have a more restricted field of view than that of humans. Whereas people can see more than 180 degrees, a typical camera might only see 90 degrees, thus missing a lot of what is going on in the periphery. A robot with a single camera will have a limited field of view and might have to rely on other sensors, such as laser range finders or microphones, to give it a sense of what is going on around it.

Most importantly, the camera image needs to be processed using computer-vision algorithms in order for the robot to be able to respond to its visual environment (see Section 3.5.4).

Depth sensors

Just as human vision uses stereo vision, knowledge about objects, and self-motion to figure out the distance to objects, so can computer-vision algorithms be used to extract a three-dimensional (3D) image from two-dimensional (2D) information. Stereo cameras have been the technology of choice for a long time, but in recent years, technologies have emerged that allow us to see depth directly, without the need for computer vision. These "depth sensors" output a "depth image" or RGBD image (with D standing for *depth*), a map of distances to objects in view of the camera.

Typically, a depth sensor can measure the distance to objects a few meters away. Depending on the strength of the emitted infrared light, most depth sensors only work reliably indoors. There are several ways of making such depth sensors. One of the typical mechanisms is time of flight (TOF), in which a device transmits invisible infrared light pulses and measures the time taken between the moment when it transmitted the light and the moment when it received the light's reflection. Because the speed of light is so high, the camera would need to record the timing of the returning light with a precision that is out of reach of current electronics hardware. Instead, the camera emits pulses of infrared light and measures the phase difference between the light leaving the camera and the light returning to the camera. The Microsoft Kinect One, the second iteration of Microsoft's game controller, is based on this principle (see Figure 3.5). Despite being developed as a game controller, it was quickly adopted by robot builders and is now widely used to give robots a sense of depth. Combined with appropriate software, the Kinect sensor can also perform skeleton tracking, which is helpful for figuring out where people are, what they are doing, and even how they are feeling. Smaller devices are now available that return RGBD images based on a range of different technologies, including TOF, structured light, and stereo vision.

This material has been published by Cambridge University Press as Human Robot Interaction by
Christoph Bartneck, Tony Belpaeime, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Sabanovic.
ISBN: 9781108735407 (http://www.cambridge.org/9781108735407).

Figure 3.5 The Microsoft Kinect Azure DK for Windows sensor. (Source: Used with permission from Microsoft)

Laser range finders

In order to measure distances at longer ranges, researchers frequently use a laser range finder, also known as light detection and ranging (LIDAR). A typical laser range finder can measure distances to objects up to 30 meters away, and it samples the environment between 10 and 50 times per second. The accuracy of laser range finders is within a few centimeters. The basic mechanism of this type of sensor is also TOF. A laser range finder transmits a single beam of infrared laser light and measures the distance by measuring the time between the moment it transmits the laser beam and the time it receives its reflection. Typically, the transmitter and receiver are on a rotating platform, sweeping the laser beam around the environment. Thus, the device only measures distance in a single 2D plane (i.e., the plane of rotation of the rotating platform).

Robots can have range finders mounted at different heights to scan for objects on a horizontal plane. Range finders close to the ground can sense objects on the floor and people's legs, whereas range finders that are set higher up can be used to sense objects on a table or counter (see Figure 3.6)

3.3.2 Audio

Microphones are commonly used devices for auditory sensing. A microphone converts sound into electrical signals. Microphones have different sensitivity profiles; some are omnidirectional, picking up all sounds in the environment, whereas others are directional, only picking up sounds in a cone-shaped area in front of the microphone. Combining multiple microphones into an array allows us to use "beam-forming" techniques, which can separate sound signals coming from a specific direction from

This material has been published by Cambridge University Press as Human Robot Interaction by
Christoph Bartneck, Tony Belpaeime, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Sabanovic.
ISBN: 9781108735407 (http://www.cambridge.org/9781108735407).

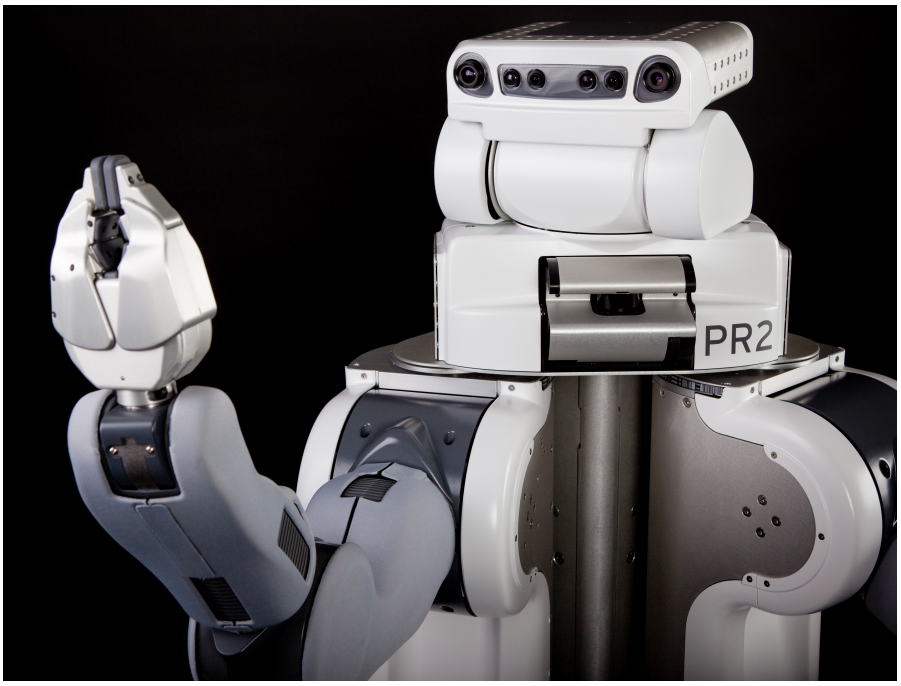

Figure 3.6 The PR2 robots (2010–2014): Can you tell where the range finder is? (Source: Willow Garage)

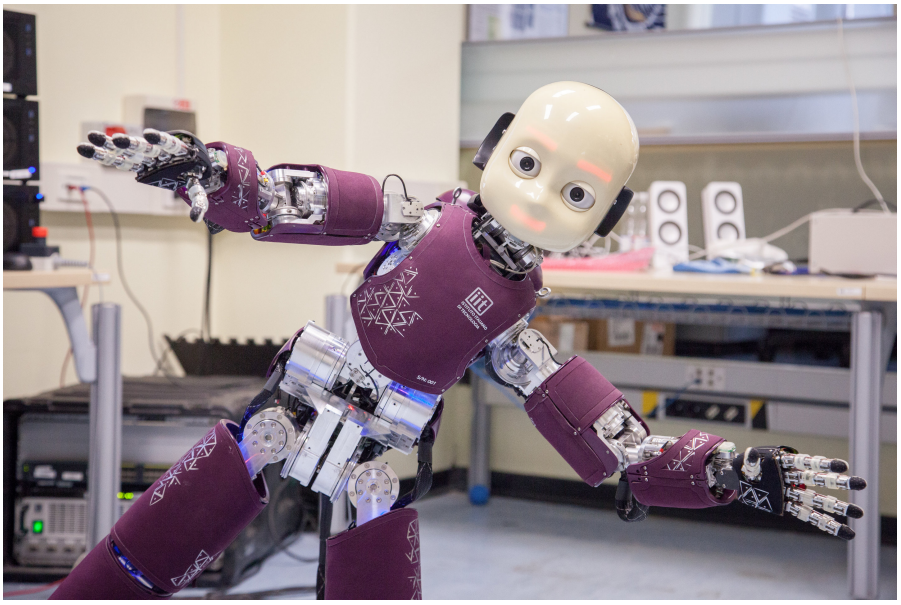

Figure 3.7 The iCub (2004–present) humanoid has capacitive tactile sensors worked into its fingers, palms, and torso. (Source: IIT Central Research Lab Genova)

ambient noises. Microphone arrays are used for sound source localization, that is, getting an accurate reading on the angle of a given sound source with respect to its position in relation to the microphone array.

3.3.3 Tactile sensors

Tactile sensors can be important in HRI, for example, when the robot is physically guided by the user. Many different implementations exist, from physical buttons or switches to capacitive sensors such as those found on touch screens (see Figure 3.7).

The most commonly used tactile sensor is a mechanical push switch. It is often used together with a bumper. When a robot collides with an object, the switch is closed, allowing the robot to detect the collision. Pressure sensors and capacity sensors, like the ones reading your finger's position on a touch screen, can also be used to detect physical contact with the environment. Pressure sensors can be implemented using a range of technologies but usually contain a material that changes its electrical properties (resistance or capacitance) when force is applied (see Figure 3.7). Pressure sensors can help robots recognize whether and how hard they are touching a person or object. They are also very useful for enabling robots to pick up and handle objects appropriately. Tactile sensors can also be used to allow the robot to know whether someone is touching it, and the robot can be programmed to respond accordingly. For example, the seal-like Paro robot has a tactile sensor net all over its body that allows it to sense the location and pressure with which a person is touching it and react by cooing for soft strokes and crying out after a harder hit.

3.3.4 Other sensors

Various other sensors exist, many of which can be relevant to HRI. Light sensors read the amount of light falling on the sensor and can be used to sense a sudden change in light, signaling that something has changed in the environment. When combined with a light source, they can be used to detect objects. A simple and very effective obstacle sensor combines an infrared light-emitting diode (LED) light with an infrared light sensor; when light bounces back from objects in front of the sensor, it can determine the distance to objects. This not only is used to detect obstacles in front of the robot but can also be used to sense when people are approaching the robot.

In recent years, the inertial measurement unit (IMU) has become a popular sensor. It combines three sensors—an accelerometer, a gyroscope, and a magnetometer—and is used the read the rotation and motion of the sensor or, more accurately, the rotational and translational acceleration. Recent advances in micro-electrical manufacturing have allowed these sensors to be miniaturized down to a few millimeters. They have become ubiquitous in mobile phones and miniature drones, and when used in a robot, they allow the robot to sense if it falls or to keep track of where it has moved over time.

This material has been published by Cambridge University Press as Human Robot Interaction by
Christoph Bartneck, Tony Belpaeime, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Sabanovic.
ISBN: 9781108735407 (http://www.cambridge.org/9781108735407).

Far infrared sensors (FIRs) are cameras that are sensitive to long-wavelength infrared light, which is emitted by warm bodies. They can be used to detect the presence of people, as used in burglar alarms, or when integrated into an FIR camera, they can be used to record an image of the temperature of the room. FIR sensors are still expensive and are mainly used for thermal imaging, but eventually, they may allow the robot to see people at night or in cluttered environments.

It is important to realize that, unlike our own senses, sensors do not necessarily need to be mounted on the robot. A robot might rely on a ceiling-mounted camera to interpret the social environment, or it could use a wall-mounted microphone array to localize who is speaking. The whole environment could, in a sense, be considered part of a robot system.

3.4 Actuators

An actuator converts electrical signals into physical movements. A system with one actuator typically realizes motion either on one straight line or on one rotational axis. This means that the system has one degree of freedom. By combining multiple motors, we can develop a robot that has motion with multiple degrees of freedom, allowing for navigation of a 2D plane or gesturing with human-like arms.

3.4.1 Motors

The standard actuator for robots is a direct-current (DC) servo motor (see Figure 3.8). It typically consists of a DC motor and a microcontroller, with a sensor such as a potentiometer or an encoder, which outputs the absolute or relative position of the motor's output axis. To control the speed, the controller typically sends pulse-width modulation (PWM) signals to the DC motor. PWM is an on/off pulse, literally switching the motor on for a few milliseconds and then back off. This is done several times per second (up to 100 times per second), and the duration of the on phase against the off phase (known as the duty cycle) determines the speed at which the motor rotates. The PWM signal controls the speed of the motor, and the controller sets the position of the motor. This is done through feedback control, where the controller continuously reads the position of the motor and adjusts the motor's PWM and direction to reach or maintain a desired position. For motors used in a robot's arms and head, the controller typically performs position control to rotate the motor toward a given commanded angle. For motors used in wheels on a mobile base, the controller typically performs velocity control to rotate the motor at the commanded velocity.

This material has been published by Cambridge University Press as Human Robot Interaction by
Christoph Bartneck, Tony Belpaeime, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Sabanovic.
ISBN: 9781108735407 (http://www.cambridge.org/9781108735407).

Figure 3.8
Connecting servo motors to each other allows robots to move around in various ways, such as in this robot arm. (Source: Trossen Robotics)

Robots can have different configurations and numbers of motors depending on the body shape and the functions they are meant to perform. Commercially available cleaning robots, such as Roomba, typically have two motors driving the wheels and one tactile sensor for moving around the room. Thus, Roomba has two degrees of freedom (DOFs). A simple nodding robot may have one motor to control its head direction, meaning that it has one DOF. A better-equipped humanoid may have three DOFs for its head, controlling pan, tilt, and yaw; two arms with four to seven DOFs; a mobile base with at least two motors; and sensors for visual, auditory, and tactile sensing. A robot arm, such as the KUKA (see Figure 3.9), must have at least six DOFs to manipulate an object. Three DOFs are necessary to locate its end effector (e.g., hand) to be in a position within a reachable range of the object, and another three DOFs are needed to grasp the object from any direction. A human arm can be approximated as an arm having seven DOFs, with an additional redundant one DOF beyond the necessary six DOFs for manipulation. To grasp objects, a robot arm must have some type of end effector attached at the end. A 1-DOF gripper can be used to grasp an object, but more complex robot hands can have as many as 16 DOFs. Android robots, designed to closely resemble humans, typically have many more DOFs (e.g., 50 DOFs) and are able to control their facial expressions and other bodily movements in relatively nuanced ways compared to simpler robots.

Motors come in many different sizes, speeds, and strengths and thus have differing power needs. It is therefore important to consider from

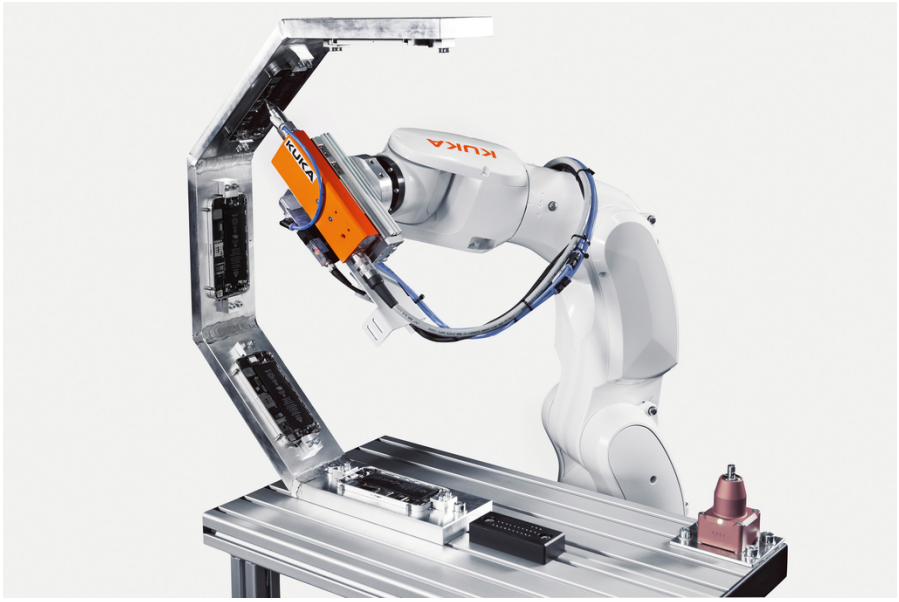

Figure 3.9 Kuka robot arm. (Source: Kuka)

early on in the design process how the motor specifications relate to the robot's design and what kinds of actions a robot will need to make, such as whether it will need to pick up a 1-kilogram bag or just needs to wave its arms; how big the robot can be while still fitting in well with its environment; how quickly it needs to respond to stimuli; and whether it needs to have a portable power bank or can be plugged into the wall.

3.4.2 Pneumatic actuators

A pneumatic actuator uses a piston and compressed air. Air is delivered from a compressor or from a vessel containing high-pressure air, which needs to be attached to the robot in some way. Pistons typically can extend and contract, depending on which valves are opened to let in the compressed air. As opposed to electric motors, pneumatic actuators produce linear motion, which is somewhat similar to human muscle motion, and are able to produce accelerations and speeds that are difficult to achieve using electric motors. Hence, they are often preferred for humanoid robots and android robots that need to gesticulate at humanlike acceleration and velocity (see Figure 3.10). The compressors that they need to operate can be quite loud, so it is important to consider how to give the robot access to compressed air without marring the interaction experience.

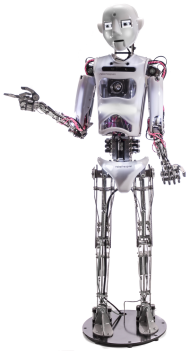

Figure 3.10 RoboThespian (2005–present) uses pneumatic actuators to achieve the acceleration required to deliver a convincing theatrical performance. The robot can run for around a day on a scuba tank's worth of compressed gas, although it can also be attached to a compressor. (Source: Photo copyright Engineered Arts)

This material has been published by Cambridge University Press as Human Robot Interaction by
Christoph Bartneck, Tony Belpaeime, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Sabanovic.
ISBN: 9781108735407 (http://www.cambridge.org/9781108735407).

3.4.3 Speakers

To generate sounds and speech, standard loudspeakers are used. Speakers are perhaps the cheapest actuator on the robot, but in terms of HRI, they are indispensable. Where to place a speaker or speakers in the robot's body is an important factor to consider when designing a robot that will interact with people. Takayama (2008) showed that the relative height from which the voices of a user and an agent interacting with each other are projected can influence who is seen to be dominant in the interaction.

3.5 Software

All the currently available robots are controlled by software running on one or several computers. The computers receive data from sensors and periodically send commands to the actuators. Some robots do all processing on-board, but many robots will offload processing to other computers. In more recent robot software, the speech recognition, computer vision, and storage of user data often happen in the cloud, transmitted by internet-connected software services, typically operating on a pay-per-use basis. The advantage of cloud-based computing is that the robot has access to much more computing power and storage than it could ever carry on-board. Smart speakers, such as Google Home and Amazon Alexa, rely on cloud-based computing. However, a disadvantage is that when a robot relies on cloud-based computing, it needs robust communication with the cloud server, which is not necessarily guaranteed, particularly when a robot is mobile. Thus, time-critical computing and computing used to guarantee safety (e.g., emergency stop) are usually done on-board.

3.5.1 Software architecture

A robot is much more than a computer with a body. A computer operates in a clean, digital environment, whereas a robot needs to interface with the messy, buzzing confusion of the real world. Not only does it need to make sense of the world, but it also needs to do so in real-time. This environment requires a radically different approach to robot software.

Architecture models

How should software for a robot be organized? A first rule of thumb, which is applicable to nonrobot software as well, is that messy program code should be avoided. Researchers and developers must aim to modularize software. One typical approach is to follow the "sense-plan-act"

This material has been published by Cambridge University Press as Human Robot Interaction by
Christoph Bartneck, Tony Belpaeime, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Sabanovic.
ISBN: 9781108735407 (http://www.cambridge.org/9781108735407).

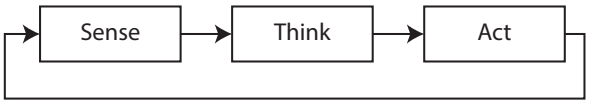

Figure 3.11
Sense-plan-act model.

model (see Figure 3.11), in which inputs from sensors are processed using software modules specific to perception, which then convert sensor streams into high-order presentations. For example, audio recordings of speech are converted into a text transcription, or camera images are analyzed to report on the location of faces. Next, there is a section that deals with "planning," which plans the robot's next actions using information gleaned from the sensing process, then outputs commands to modules for action.

For instance, a person-finding perception module reports on the location of people detected in a 2D camera image and also returns the size of the heads, indicative of how close people are to the robot. Next, the planning module computes the head orientation for the robot to face the nearest speaker and sends a command to move the head to the output modules. The output modules then calculate which angle is needed for the robot's neck motors and send these to the low-level motor controllers.

The sense-plan-act approach is also known as the *deliberative approach* because the robot deliberates its next action. Quite often, we want a robot to respond quickly to external events, without spending a lot of time pondering what to do next. In this case, we often program simple "behaviors" for the robot (Brooks, 1991). Behaviors are tightly coupled sensor–action processing loops, which immediately respond to an external event. These can be used to make an emergency stop when the robot is about to drive down the stairs, but they can serve equally well in social interaction. When a loud bang is heard, or when a face appears in view, we want the robot to respond as fast as possible. Act first; think later. Often, there are dozens of behaviors running on the robot, and mechanisms exist to mediate between which behaviors are active and which are not. One such mechanism is the subsumption architecture, which organizes behavior into hierarchies, allowing a behavior to activate or inhibit others (Brooks, 1986) (see Figure 3.12).

With this approach, even though the robot does not have an explicit "representation" of the world, it can still behave in an apparently intelligent way. For instance, if a cleaning robot uses two behaviors in parallel, one that avoids the wall and another that makes it have a slight pull to the right, the resulting, or emergent, behavior is that of wall following. Even though wall following wasn't programmed explicitly, it emerges from the interaction between two simpler behaviors. The vacuum robot Roomba has been developed with such an idea in mind.

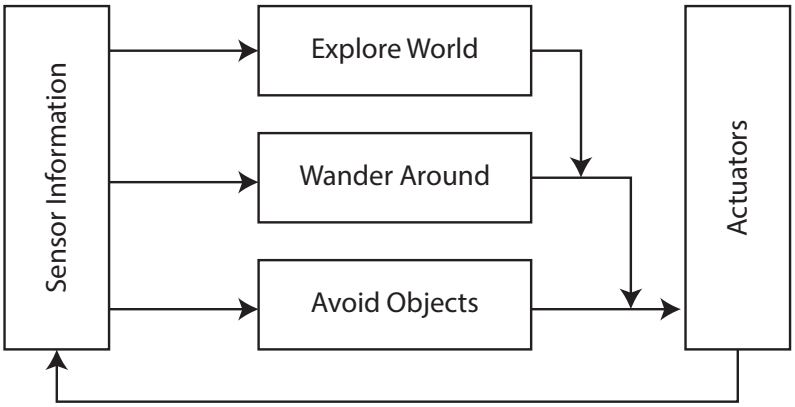

Figure 3.12 The subsumption behavior-based architecture.

In HRI studies, we typically find ourselves looking for a middle ground between the deliberative and reactive approaches. We want a reactive control layer, which responds quickly to subsecond social events, followed by a deliberative layer, which formulates a coherent response to slower elements of the interaction, such as conversation.

In light of this, it is important to develop software that can be decomposed into a number of smaller modules. Even if the complete wealth of a sense-plan-act model is not needed, it is still common practice to separate modules into perception, planning, and action.

Planning is diverse in terms of components and complexity and depends heavily on the robot and the application. A cleaning robot may need to compute the next location to clean, whereas a companion robot may need to make a decision on how it should initiate a conversation with a user. The software on a Roomba vacuum will therefore be radically different from that on a Pepper humanoid. For interactive robots, various forms of HRI knowledge will be embedded into the various software modules.

Action modules take care of the actuation and social output of the robot, such as nonverbal utterances, speech, hand gestures, and locomotion. For instance, the speech-synthesis module may receive text and convert this into spoken words together with timing information that allows the robot to accentuate its speech with appropriate gestures.

3.5.2 Software-implementation platform

Software typically runs on an operating system (e.g., Windows or Linux) and typically on some implementation platform. Robot Operating System (ROS) is a platform commonly used in the robotics and HRI communities. It deals with communications between sensors and modules and offers libraries and tools to support frequently used robot abili-

This material has been published by Cambridge University Press as Human Robot Interaction by
Christoph Bartneck, Tony Belpaeime, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Sabanovic.
ISBN: 9781108735407 (http://www.cambridge.org/9781108735407).

ties, such as localization and navigation. ROS has a large community of users, who often share modules on public software repositories.

3.5.3 Machine learning

Some tasks can be learned rather than being programmed explicitly. The practice of letting a robot learn a skill is called *machine learning*. There are various machine-learning techniques, as described next.

Training data

Machine learning requires data from which the robot can learn. This training data set should contain a large number of examples of the thing to be learned, which may be data from sensors or text and generally has been manually annotated by people. For instance, there can be a data set with camera images of human faces, and for each image, the emotion of the person is labeled, such as "neutral," "smile," or "angry." Typical data sets contain hundreds of thousands or even millions of examples.

Feature extraction

To aid machine learning, sensor data are often preprocessed by converting the sensor data into a more suitable representation and by extracting salient features from the data. This process is called feature extraction. There are many algorithms to extract features from raw sensor input. For instance, edge detection highlights the pixels in an image where the intensity abruptly changes, and a segmentation algorithm identifies regions in an image where the colors are all similar, which can indicate a face, hair, or an eye (see Figure 3.13).

Figure 3.13
Canny edge detection of a user operating the buttons on a robot.

Features are, in essence, numbers. Often these features are placed into a feature vector, a row of numbers ready for processing. For instance, one could count up the number of pixels detected as an edge and use it as one of the variables of the feature vector. Researchers often manually analyze their data sets and identify salient features. For instance, with careful observation, one might find that a child fidgets more than an adult does; once such a feature is found, one can add variation of motion to the feature vector.

Classification based on training

There are a number of machine-learning approaches. One often-used approach is *classification*. In classification, an algorithm decides, based on training data, what class an unknown data point belongs to. For example, given a camera image of a person, the classifier decides what emotion the person's face shows.

Suppose we can compute a one-dimensional (1D) feature vector representing people's height and have a data set with two classes, "child"

or "adult" (i.e., each data point in the training data will have a label saying whether the data point is a "child" or "adult"). The classifier learns a threshold value from the training data set (e.g., 150 cm) to distinguish the two classes.

In this case, the feature vector contains only a single feature, the height of the user. We call this a 1D feature vector. Machine-learning algorithms typically work with thousands of features and try to recognize up to thousands of classes. Classification errors often happen. For instance, a tall child or short adult would probably be classified incorrectly.

Machine-learning algorithms perform better when having access to more data. Ideally, we want machine-learning algorithms to "generalize," meaning they correctly handle data that they have never been exposed to. However, sometimes machine learning produces an algorithm that "overfits." When this happens, the algorithm does really well on the data it has been trained on, but it performs poorly when confronted with new problems.

Deep learning

Deep learning, also known as deep neural networks (DNNs), is a machine-learning technique made possible through the increased availability of computational power. Deep learning relies on artificial neural networks with a large number of layers of interconnected artificial neurons—hence the name "deep." It takes a large amount of computational power to train DNNs, but recent progress in using parallel computing and graphical-processing units (GPUs) has allowed us to train these networks within a matter of days.

DNNs do not require careful feature extraction by hand. Instead, DNNs discover the relevant features from the data by themselves. A drawback is that DNNs require huge amounts of data, typically millions of data points. For instance, Google collected an enormous data set, containing more than 230 billion data points, to train its speech-recognition algorithm.

The need for large data sets is a significant challenge for HRI because it is difficult to collect large amounts of data in which humans and robots are interacting. The complexity of deep learning also makes it difficult to know exactly what the network bases its decisions on (e.g., we may not know what features it has identified or how it decided to use these features to come to a classification), which can be particularly problematic for HRI outside of the laboratory when we need to trust that the system will be robust, safe, and predictable. If the robot does something wrong, we need to be able to figure out how to debug and correct the system, as in the case of an autonomous Uber vehicle that had trouble classifying a person crossing the road and ran over the person as a result (Marshall and Davies, 2018).

3.5.4 Computer vision

Computer vision is an important area for HRI. In essence, computer vision interprets a 2D array of numbers when working with single images, or a series of 2D images recorded over a period of time when working with video data. Computer vision can be rather straightforward and still very effective in the context of HRI. Motion detection, for example, can be achieved by subtracting two camera images taken just a fraction of a second apart. Any pixels that captured motion will have a nonzero value, which in turn can be used to calculate the region with the most motion. When used on a robot, a motion detector lets the robot orient itself toward the areas with the most motion, providing the illusion that the robot is aware of things moving, which, in the context of HRI, often involves people gesturing or talking.

Another computer-vision technique relevant to HRI is processing faces. The ability to detect faces in an image has advanced and can be used, for example, to let the robot look people in the eye. Face recognition (i.e., identifying a specific person in an image) is still a challenge, however. Impressive progress has been made in recent years, mainly fueled by the evolution of deep learning, and it is now possible to reliably recognize and distinguish between hundreds of people when they are facing the camera. But face recognition typically fails when the user is seen from the side.

Skeleton tracking is another technique relevant to HRI. In skeleton tracking, the software attempts to track where the user's body and limbs are. This technique was first used in gaming on the Microsoft Xbox console, with software specific to the Kinect RGBD sensor, but is now a staple in many HRI applications. Several software solutions exist, but recently, deep learning has enabled the reading of skeletons of dozens of users in complex scenarios from a single simple camera image, without the need for an RGBD sensor. The software for this, called OpenPose, is now freely available and often used in HRI studies (Cao et al., 2017).

There are many commercial and free software solutions that offer a range of out-of-the-box computer-vision functionality. OpenCV is perhaps the best-known offering; it is a free software library, developed over 20 years, that can be used for facial recognition, gesture recognition, motion understanding, object identification, depth perception, and motion tracking, among others.

Because computer vision often requires a considerable amount of computational power, which is not realistic on small or cheaper robots, sometimes the computer-vision process is addressed on the cloud. In this case, the video stream of the robot is sent over an internet connection to servers on the cloud. There are commercial-based cloud solu-

This material has been published by Cambridge University Press as Human Robot Interaction by
Christoph Bartneck, Tony Belpaeime, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Sabanovic.
ISBN: 9781108735407 (http://www.cambridge.org/9781108735407).

tions for face recognition, person identification, and image classification being sold on a per-use basis.

3.6 Limitations of robotics for HRI

There are several limitations of robotics, some of which are specific to HRI and some of which apply to robotics in general. One general challenge is that a robot is a complex system that needs to translate between the analogue world and the digital internal computation of the robot. The real world is analogue, noisy, and often very changeable, and the robot first needs a suitable digital representation of the world, which the software then uses to make decisions. Once a decision is made, this is translated back into analogue actuation, such as speaking a sentence or moving a leg.

Another major challenge applicable to all of robotics is that of learning. Currently, machine learning needs to iterate through millions of examples to slowly nudge itself toward performing a task with a reasonable level of skill. Despite speedups due to advances in DNNs and GPUs, at the time of writing, computers need days or often weeks to learn, and this is only when all the learning can happen internally, for example, in simulation or using prerecorded data. Learning from real-time data that a robot samples from the world is still virtually impossible. Related to this is the challenge of "transfer," or the performance of one skill transferring to another. For example, people can learn to play one game of cards and will then be able to transfer that knowledge to quickly pick up another game of cards with different rules. Machine learning typically struggles with this task and needs to start the learning of a new challenge from scratch.

The seamless integration of the various systems on a robot is also a major challenge. Speech recognition, natural-language understanding, social-signal processing, action selection, navigation, and many other systems all need to work together in order to create convincing social behavior in a robot. On simple robots, this is manageable, but on more complex robots, the integration and synchronization of these various skills are still beyond our grasp. Face detection, emotion classification, and sound-source localization might each work well in isolation, but bringing the three together to make the robot respond in a humanlike manner to people approaching the robot is still a challenge. Greeting people who smile at the robot, looking up when the door slams, or ignoring people who show no interest in the robot sound easy, but it is difficult to build such behavior that consistently works well. The challenge becomes formidable once further skills are added. Conversational robots, which aim to interact with people using natural language in addition to using their full suite of sensors to react in an appropriate manner, are only now being attempted in research labs across the

world. It is unlikely that a robot will be built in the next decade that can handle a conversation as well as people can.

Robots and artificial intelligence (AI) systems in general struggle with semantics: they often do not truly understand what happens around them. A robot might seem to respond well to a person approaching it and asking for directions, but this does not mean that the robot understands what is happening—that the person is new to the space, or where the directions it gives actually lead to. Often, the robot has been programmed to face people when they come near and to respond to key words it hears. Real understanding is, at the moment, still exclusive to humans. Although there are research projects on imbuing AI systems with a sense of understanding (Lenat, 1995; Navigli and Ponzetto, 2012), there are not yet robots that can use their multimodal interaction with the world to understand the social and physical environment.

The reasons why AI has not yet achieved a humanlike general intelligence level are manifold, but conceptual problems were identified right from the outset. Searle (1980) pointed out that digital computers alone can never truly understand reality because they only manipulate syntactical symbols that do not contain semantics. In his *Chinese Room thought experiment*, a slip of paper with Chinese symbols is slid under the door of a room. A man inside the room reads the symbols and comes up with a response by applying a set of rules he finds in a book full of instructions containing more Chinese characters. He then writes the response in the form of other Chinese characters and slides it back under the door. The audience behind the door might be under the impression that the man in the room understands Chinese, whereas in reality, he just looks up rules and has no understanding of what those symbols really mean. In the same manner, a computer also only manipulates symbols to come up with a response to input. If the computer's response is of humanlike quality, does that mean the computer is intelligent?

According to Searle's line of argument, IBM's chess-playing computer Deep Blue does not actually understand chess, and DeepMind's AlphaGo does not understand the game of Go. Both programs may have beaten human masters of the game, but they did so only by manipulating symbols that were meaningless to them. The creator of Deep Blue, Drew McDermott, replied to this criticism: "Saying Deep Blue doesn't really think about chess is like saying an aeroplane doesn't really fly because it doesn't flap its wings" (1997). That is, he debated that as far as it functions as it is supposed to, a new machine or AI does not need to replicate all the details of humans, animals, or birds. This debate reflects

different philosophical viewpoints about what it means to think and understand and is still under way today. Similarly, the possibility of developing general AI remains an open question. All the same, progress has been made. In the past, a chess- or Go-playing machine would have been regarded as intelligent. But now it is regarded as the feat of a calculating machine—our criteria for what constitutes an intelligent machine have shifted along with the capabilities of machines.

In any case, no sufficiently intelligent machine has yet been built that would provide a foundation for many of the advanced application scenarios that have been imagined for robots. Researchers often fake the intelligence of the robot by applying the Wizard-of-Oz method (see p. 156). The requirements of HRI often imply unrealistic assumptions about what can be achieved with current technology, and novice research and the public should be aware of the limitations of robotics and AI.

3.7 Conclusion

Robots are made from multiple software modules connected with sensors and actuators. Software design requires HRI knowledge, and conversely, HRI researchers need to have a basic understanding of software in order to provide useful knowledge for future HRI developers. For a robot to be successful, the different components need to be chosen and integrated with an eye toward the specific HRI application and its needs. Despite limitations, however, robots can be designed to interact successfully with humans in various types of short-term, and sometimes longer, interactions.

Questions for you to think about:

- Chapters 2 and 3 introduce various robot types that are available on the market. What sensors do these robots have? What actuators do they have? What hardware components do you think are crucial?
- Imagine a scenario where you want to use a smart social robot. Which sensors and actuators should it have? What skills should the robot have, and is software available to deliver these skills?
- What kind of data set would be needed to train a machine-learning algorithm for a new interaction capability of a robot, such as distinguishing your face from others?

This material has been published by Cambridge University Press as Human Robot Interaction by
Christoph Bartneck, Tony Belpaeime, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Sabanovic.
ISBN: 9781108735407 (http://www.cambridge.org/9781108735407).

Future reading:

- For basic AI:
  Stuart Russell and Peter Norvig. *Artificial intelligence: A modern approach*. Pearson, Essex, UK, 3rd edition, 2009. ISBN 978-0136042594. URL `http://www.worldcat.org/oclc/496976145`
- For basic robotics:
  Maja J. Matarić. *The robotics primer*. MIT Press, Cambridge, MA, 2007. ISBN 9780262633543. URL `http://www.worldcat.org/oclc/604083625`
- For diverse topics in robotics:
  Bruno Siciliano and Oussama Khatib. *Springer handbook of robotics*. Springer, Berlin, 2016. ISBN 9783319325507. URL `http://www.worldcat.org/oclc/945745190`