http://www.robotshop.com/blog/en/how-to-make-a-robot-lesson-1-3707

How to Make a Robot - Lesson 1: Getting Started - RobotShop Blog

Lessons Menu: Lesson 1 – Getting Started Lesson 2 – Choosing a Robotic Platform Lesson 3 – Making Sense of Actuators Lesson 4 – Understanding Microcontrollers Lesson 5 – Choosing a Motor Controller Lesson 6 – Controlling your Robot Lesson 7 – Using Sensors Lesson 8 – Getting the Right Tools Lesson 9 – Assembling a Robot Lesson 10 – Programming a Robot Getting Started Welcome [...]
Read more...

Remove preview

How to Make a Robot – Lesson 1: Getting Started

Posted on August 4, 2010 by Coleman Benson & filed under How To Make a Robot, Tutorials.

Lessons Menu:

Getting Started

Welcome to the first installment of the Grand RobotShop Tutorial, a series of 10 lessons that will teach you how to make your own robot. This tutorial is aimed at anybody willing to get started in robotics and have a basic understanding of terms such as "voltage", "current", "motor", and "sensors". Although this might seem pretty basic, even people with previous robot building experience might find useful information regarding the general method of building a robot.

What is a robot?

There are many definitions of robot and no real consensus has been attained so far. We loosely define a robot as follows:

Robot: An electromechanical device which is capable of reacting in some way to its environment, and take autonomous decisions or actions in order to achieve a specific task.

This means that a toaster, a lamp, or a car would not be considered as robots since they have no way of perceiving their environment. On the other hand, a vacuum cleaner that can navigate around a room, or a solar panel that seeks the sun, can be considered as a robotic system.

It is also important to note that the "*robots*" featured in *Robot Wars* for instance or any solely remote controlled device would not fall under this definition and would be closer to a more complex remote controlled car.

Although this definition is quite general, it might need to evolve in the future in order to keep up with the latest advancement in the field. In order to get a sens of how robotics is rapidly growing, we suggest you take a look at the RobotShop History of Robotics.

Let's get started

This series of tutorials is intended to guide you through the steps of building a complete mobile robot.

There are **10 lessons** that will be released in the following **10 weeks**. Each lesson guides you through one step of making a general-purpose mobile robot. This will enable you to build your very own mobile robot in order to perform a task of your choice. Each lesson will be illustrated with an example from RobotShop experience in producing the RobotShop Rover. The lessons are intended to be read one after the other and build upon the information gained.

STEP 1

The first step is to determine what your robot should do (i.e. **what is its purpose in life**). Robots can be used in almost any situation and are primarily intended to help humans in some way. If you are unsure of what you want your robot to do or simply want to concentrate your efforts on specific tasks, here are some ideas:

Knowledge & Learning

In order to build increasingly complex robots, most professionals and hobbyists use knowledge they have acquired when building previous robots. Instead of building one robot, you can learn how to use individual components with the objective of building your own "*knowledge library*" to use to undertake a larger, more complex design in the future.

Amusement & Companionship

Building a robot is in and of itself is fun and exciting. Robotics incorporates aspects of many disciplines including engineering (mechanical, electrical, computer), sciences (mathematics and physics) and arts (aesthetics) and users are free to use their imagination. Amusing others with your creations (especially if they are user-friendly and interactive) helps others to become interested in the field.

Competitions & Contests

Competitions give the project design guidelines and a due date. They also put your robot against others in the same class and test your design and construction skills. Although

many competitions are specifically for students (elementary to university), there also exist open competitions where adults and professionals alike can compete.

Autonomous life form


Humans are natural creators and innovators. The next great innovation will be to develop a fully autonomous life form that rivals or surpasses ourselves in ability and perhaps creativity. This goal is still being accomplished in small steps by individuals, research organizations and professionals.

Domestic or Professional tasks


Domestic robots help liberate people from unpleasant or dangerous tasks and give them more liberty and security. Professional and Service Robots are used in a variety of applications at work, in public, in hazardous environments, in locations such as deep-sea, battlefields and space, just to name a few. In addition to the service areas such as cleaning, surveillance, inspection and maintenance, we utilize these robots where manual task execution is dangerous, impossible or unacceptable. Professional and Service Robots are more capable, rugged and often more expensive than domestic robots and are ideally suited for professional and/or commercial use.

Security and Surveillance

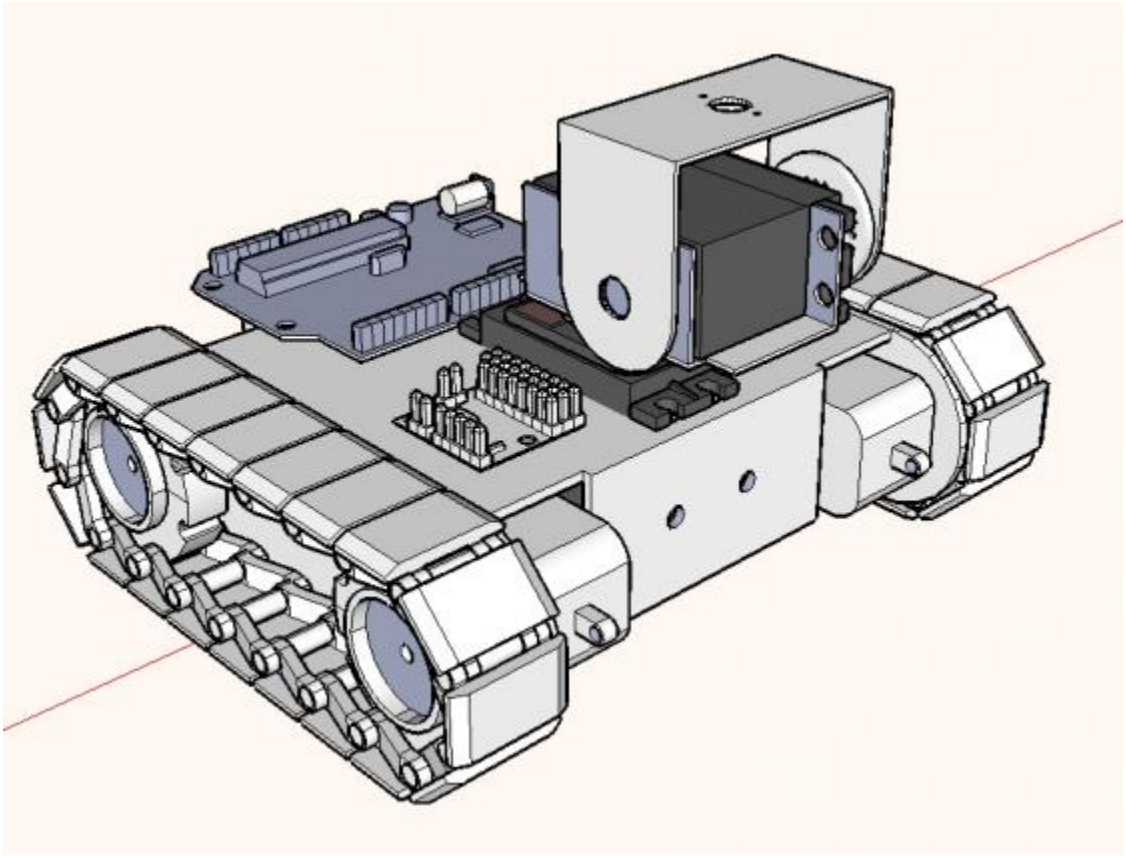

Most mobile robots are used to venture into areas where humans either should not or cannot go. Robots of various sizes (either remote controlled, semi-autonomous or fully autonomous) are an ideal choice for these tasks.

Practical Example

We anticipate that most of you following this guide have the objective of building a robot for learning and knowledge, but also for sheer fun; though many will have a specific idea or project they want to materialize.

The last major consideration is budget. It is difficult to know exactly what people have in mind when they build their first robot; one might already want to build an autonomous snow removal robot, while another simply wants to make an intelligent clock. A simple programmable mobile robot might cost about $100 while a more complex can be several thousands of dollars.

In this exercise, we have chosen to make a mobile platform in order to get an understanding of motors, sensors, microcontrollers and programming, and to include a variety of sensors. We'll keep the budget to about $200 to $300 since we want it to be fairly complete.



For further information on learning how to make a robot, please visit the RobotShop Learning Center. Visit the RobotShop Community Forum in order to seek assistance in building robots, showcase your projects or simply hang-out with other fellow roboticists.

Tags: Grand Tutorial Series Make a Robot RobotShop Tutorial

Lesson 2

Lessons Menu:

Following the first lesson, you now have a basic understanding of what a robot is and what current robots normally do.

Now, it is time to decide on the type if robot you are going to build. A custom robot design often starts with a "vision" of what the robot will look like and what it will do. The types of robots possible are unlimited, though the more popular are:

- Land wheeled, tracked, and legged robots
- Aerial planes, helicopters, and blimp
- Aquatic boats, submarines, and swimming robots
- Misc. and mixed robots

- Stationary robot arms, and manipulators

This lesson is intended to help you decide what type of robot to build to best suite your mission. Since you have brainstormed on what tasks or functions you want it to accomplish (after lesson 1), you can now choose the type of robot that will best suite your needs. Below, you will find a description of all the major robot types.

Land

Land-based robots, especially the wheeled ones, are the most popular mobile robots among beginners as they usually require the least investment while providing significant exposure to robotics. On the other hand, the most complex type of robots is the humanoid (akin to a human), as it requires many degrees of freedom and synchronizing the motion of many motors, and uses many sensors.

Wheeled Robots

Wheels are by far the most popular method of providing mobility to a robot and are used to propel many different sized robots and robotic platforms. Wheels can be just about any size, from a few centimetres up to 30 cm and more . Tabletop robots tend to have the smallest wheels, usually less than 5 cm in diameter. Robots can have just about any number of wheels, although 3 and 4 are the most common. Normally a three-wheeled robot uses two wheels and a caster at one end. More complex two wheeled robots may use gyroscopic stabilization. It is rare that a wheeled robot use anything but skid steering (like that of a tank). Rack and pinion steering such as that found on a car requires too many parts and its complexity and cost outweigh most of its advantages.

Four and six wheeled robots have the advantage of using multiple drive motors (one connected to each wheel) which reduces slip. Also, omni-directional wheels or mecanum wheels, used properly, can give the robot significant mobility advantages. A common misconception about building a wheeled robot is that large, low-cost DC motors can propel a medium sized robot. As we will see later in this series, there is a lot more involved than just a motor.

Advantages

- Usually low-cost compared to other methods
- Simple design and construction
- Abundance of choice

- Six wheels or more rival a track system
- Excellent choice for beginners

Disadvantages

- May lose traction (slip)
- Small contact area (only a small rectangle or line underneath each wheel is in contact with the ground)

Tracked Robots

Tracks (or treads) are what tanks use. Although tracks do not provide added "force" (torque), they do reduce slip and more evenly distribute the weight of the robot, making them useful for loose surfaces such as sand and gravel. Also, a track system with some flexibility can better conform to a bumpy surface. Finally, most people tend to agree that tank tracks add an "aggressive" look to the robot as well.

Advantages

- Constant contact with the ground prevents slipping that might occur with wheels
- Evenly distributed weight helps your robot tackle a variety of surfaces
- Can be used to significantly increase a robot's ground clearance without incorporating a larger drive wheel

Disadvantages

- When turning, there is a sideways force that acts on the ground; this can cause  damage to the surface the robot is being used on, and cause the tracks to wear
- Not many different tracks are available (robot is usually constructed around the tracks)
- Drive sprocket might significantly limit the number of motors that can be used.

- Increased mechanical complexity (idler placement and number, # of links) and connections

Legs

An increasing number of robots use legs for mobility. Legs are often preferred for robots that must navigate on very uneven terrain. Most amateur robots are designed with six legs, which allow the robot to be statically balanced (balanced at all times on 3 legs); robots with fewer legs are harder to balance. The latter require "dynamic stability", meaning that if the robot stops moving mid-stride, it might fall over. Researchers have experimented with monopod (one legged "hopping") designs, though bipeds (two legs), quadrupeds (four legs), andhexapods (six legs) are the most popular.

Advantages

- Closer to organic or natural motion
- Can potentially overcome large obstacles and navigate very rough terrain

Disadvantages

- Increased mechanical, electronic and coding complexity (not the easiest way to get into robotics).
- Lower battery size despite increased power demands
- Higher cost to build

Air

A AUAV (Autonomous Unmanned Aerial Vehicle) is very appealing and is entirely within the capability of many robot enthusiasts. However, the advantages of building an autonomous unmanned aerial vehicles, especially if you are a beginner, have yet to outweigh the risks. When

considering an aerial vehicle, most hobbyists still use existing commercial remote controlled aircraft. On the professional side, aircraft such as the US military Predator were initially semi-autonomous though in recent years Predator aircraft have flown missions autonomously.

Advantages

- Remote controlled aircraft have been in existence for decades (so there is a large community, at least for the mechanics)
- Excellent for surveillance

Disadvantages

- The entire investment can be lost in one crash.
- Limited robotic community to provide help for autonomous control

Water

An increasing number of hobbyists, institutions and companies are developing unmanned underwater vehicles. There are many obstacles yet to overcome to make underwater robots attractive to the wider robotic community though in recent years, several companies have commercialized pool cleaning "robots". Underwater vehicles can use ballast (compressed air and flooded compartments), thrusters, tail and fins or even wings to submerge. Other aquatic robots such as pool cleaners are useful commercial products.

Advantages

- Most of our planet is water, so there is a lot to explore and discover
- Design is almost guaranteed to be unique
- Can be used and/or tested in a pool

Disadvantages

- Robot can be lost many ways (sinking, leaking, entangled…)
- Most electronic parts do not like water (also consider water falling on electronics when accessing the robot after a dive)
- Surpassing depths of 10m or more can require significant research and investment
- Very limited robotic community to provide help
- Limited wireless communication options

Miscellaneous and hybrid combinations

Your idea for a robot may not fall nicely into any of the above categories or may be comprised of several different functional sections. Note again that this guide is intended for mobile robots as opposed to stationary or permanently fixed designs (other than robotic arms and grippers). It is wise to consider when building a hybrid design, to use a modular design (each functional part can be taken off and tested separately). Miscellaneous designs can include hovercraft, snake-like designs, turrets and more.

Advantages

- Designed and built to meet specific needs
- Multi-tasking and can be comprised of modules
- Can lead to increased functionality and versatility

Disadvantages

- Possible Increased complexity and cost
- Often times, parts must be custom designed and built

Arms & Grippers

Although these do not fall under the category of **mobile** robotics, the field of robotics essentially started with arms and end-effectors (devices that attach to the end of an arm such as grippers, electromagnets etc). Arms and grippers are the best way for a robot to interact with the environment it is exploring. Simple robot arms can have just one motion, while more complex arms can have a dozen or more unique degrees of freedom.

Advantages

- Very simple to very complex design possibilities
- Easy to make a 3 or 4 degree of freedom robot arm (two joints and turning base)

Disadvantages

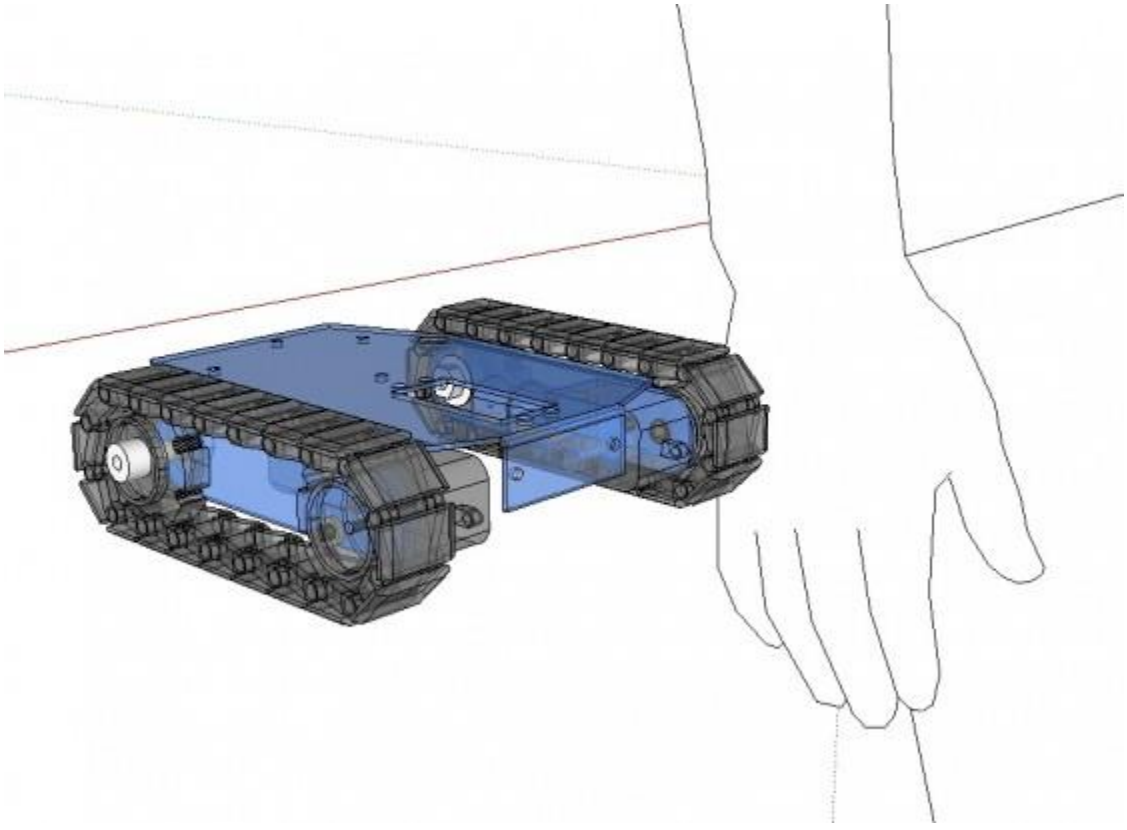
- Stationary unless mounted on a mobile platform
- Cost to build is proportional to lifting capability

Practical Example

In our case, we have opted for building a robot that will provide the maximum exposure to robotics. A programmable tracked platform that can accommodate a variety of sensors and gripper sees ideal in this case, specially since we consider tank tracks are far cooler than wheels.

In order to keep the costs down, we opted to build a small desktop robot that will be able to roam indoors and on tabletops. We also have taken into consideration the fact that there are not many tracks available, and to keep things simple, we'll only consider a single drive sprocket and single idler sprocket system, this should not be a problem since the robot will be very light weight.

The preliminary CAD below summarized the features describes so far.



Next, we will be **choosing the right actuators** (e.g. motors) for your robot.

For further information on learning how to make a robot, please visit the RobotShop Learning Center. Visit the RobotShop Community Forum in order to seek assistance in building robots, showcase your projects or simply hang-out with other fellow roboticists.

Tags: Grand Tutorial Series Make a Robot

Lesson 3

Making Sense of Actuators

Now that we learned about robotics in general in Lesson 1 and decided on the robot to make in Lesson 2, we will now choose the actuators that will make the robot move.

What is an actuator?

An “actuator” can be defined as a device that converts energy (in robotics, that energy tends to be electrical) into physical motion. The vast majority of actuators produce either rotational or linear motion. For instance, a “DC motor” is therefore a type of actuator.

Choosing the right actuators for your robot requires an understanding of what actuators are available, some imagination, and a bit of math and physics.

Rotational Actuators

As the name indicates, this type of actuators transform electrical energy into a rotating motion. There are two main mechanical parameters distinguishing them from one another: (1) torque, the force they can produce at a given distance (usually expressed in N•m or Oz•in), and (2) the rotational speed (usually measured in revolutions per minutes, or rpm).

AC Motor

 AC (alternating current) is rarely used in mobile robots since most of them are powered with direct current (DC) coming from batteries. Also, since electronic components use DC, it is more convenient to have the same type of power supply for the actuators as well. AC motors are mainly used in industrial environments where very high torque is required, or where the motors are connected to the mains / wall outlet.

DC Motors

 DC motors come in a variety of shapes and sized although most are cylindrical. They feature an output shaft which rotates at high speeds usually in the 5 000 to 10 000 rpm range. Although DC motors rotate very quickly in general, most are not *strong* (low torque). In order to reduce the speed and increase the torque, a gear can be added.

To incorporate a motor into a robot, you need to fix the body of the motor to the frame of the robot. For this reason motors often feature mounting holes which are generally located on the face of the motor so they can be mounted perpendicularly to a surface. DC motors can operate in clockwise (CW) and counter clockwise (CCW) rotation. The angular motion of the turning shaft can be measured using encoders or potentiometers.

Geared DC Motors

 A DC gear motor is a DC motor combined with a gearbox that works to decrease the motor's speed and increase the torque. For example, if a DC motor rotates at 10 000 rpm and produces 0.001 N•m of torque, adding a 256:1 ("two hundred and fifty six to one") gear down would reduce the speed by a factor of 256 (resulting in 10 000rpm / 256 = 39 rpm), and increase the torque by a factor of 256 (0.001 x 256 = 0.256 N•m). The most common types of

gearing are "spur" (the most common), "planetary" (more complex but allows for higher gear-downs in a more confined space, as well as higher efficiency) and "worm" (which allows for very high gear ratio with just a single stage, and also prevents the output shaft from moving if the motor s not powered). Just like a DC motor, a DC gear motor can also rotate CW and CCW. If you need to know the number of rotations of the motor, an "encoder" can be added to the shaft.

R/C Servo Motors

R/C (or hobby) servo motors are types of actuators that rotate to a specific angular position, and were classically used in more expensive remote controlled vehicles for steering or controlling flight surfaces. Now that they are used in a variety of applications, the price of hobby servos has gone down significantly, and the variety (different sizes, technologies, and strength) has increased.

The common factor to most servos is that the majority only rotate about 180 degrees. A hobby servo motor actually includes a DC motor, gearing, electronics and a rotary potentiometer (which, in essence, measures the angle). The electronics and potentiometer work in unison to activate the motor and stop the output shaft at a specified angle. These servos are generally have three wires: ground, voltage in, and a control pulse. The control pulse is usually generated with a servo motor controller. A "robot servo" is a new type of servo that offers both continuous rotation and position feedback. All servos can rotate CW and CCW.

Industrial Servo Motors

An industrial servo motor is controlled differently than a hobby servo motor and is more commonly found on very large machines. An industrial servo motor is usually made up of a large AC (sometimes three-phase) motor, a gear down and an encoder which provides feedback about angular position and speed. These motors are rarely used in mobile robots because of their weight, size, cost and complexity. You might find an industrial servo in a more powerful industrial robotic arm or very large robotic vehicles.

Stepper Motors

A stepper motor does exactly as its name implies; it rotates in specified "steps" (actually, specific degrees). The number of degrees the shaft rotates with each step (step size) varies based on several factors. Most stepper motors do not include gearing, so just like a DC motor, the torque is often low. Configured properly, a stepper can rotate CW and CCW and can be moved to a desired angular position. There are *unipolar* and *bipolar* stepper motor types. One notable downside to stepper motors is that if the motor is not powered, it's difficult to be certain of the motor's starting angle.

Adding gears to a stepper motor has the same effect as a adding gears to a DC motors: it increases the torque and decreases the output *angular* speed. Since the speed is reduced by the gear ratio, the step size is also reduced by that same factor. If the non geared down stepper motor had a step size of 1.2 degrees, and you add a gear down of 55:1, the new step size would be 1.2 / 55 = 0.0218 degrees.

Linear Actuators

A linear actuator produces linear motion (motion along one straight line) and have three main distinguishing mechanical characteristics: the minimum and maximum distance the rod can move "a.k.a. the "stroke", in mm or inches), their force (in Kg or lbs), and their speed (in m/s or inch/s).

DC Linear Actuator

A DC linear actuator is often made up of a DC motor connected to a lead screw. As the motor turns, so does the lead screw. A traveller on the lead screw is forced either towards or away from the motor, essentially converting the rotating motion to a linear motion. Some DC linear actuators incorporate a linear potentiometer which provides linear position feedback. In order to stop the actuator from destroying itself, many manufacturers include limit

switches at either end which cuts power to the actuator when pressed. DC linear actuators come in a wide variety of sizes, strokes and forces.

Solenoids


Solenoids are composed of a coil wound around a mobile core. When the coil is energized, the core is pushed away from the magnetic field and produces a motion in a single direction. Multiple coils or some mechanical arrangements would be required in order to provide a motion in two directions. A solenoid's stroke is usually very small but their speed is very fast. The strength depends mainly on the coil size and the current going trough it. This type of actuator is commonly used in valves or latching systems and there is usually no position feedback (it's either fully retracted or fully extended).

Muscle wire


Muscle wire is a special type of wire that will contract when an electric current traverses it. Once the current is gone (and the wire cools down) it returns to its original length. This type of actuator is not very strong, fast or provides a long stroke. Nevertheless, it is very convenient when working with very small parts or in a very confined space.

Pneumatic and Hydraulic


Pneumatic and hydraulic actuators use air or a liquid (e.g. water or oil) respectively in order to produce a linear motion. These types of actuators can have very long strokes, high force and high speed. In order to be operated they require the use of a fluid

compressor which makes them more difficult to operate than regular electrical actuators. Because of they high force speed and generally large size, they are mainly used in industrial environments.

Choosing an Actuator

To help you with the selection of an actuator for a specific task, we have developed the following questions to guide you in the right direction.

It is important to note that there are always new and innovative technologies being brought to market and nothing is set in stone. Also note that an single actuator may perform very different task in different contexts. For instance, with additional mechanics, an actuator that produces linear motion may be used to rotate an object and vice versa (like on a car's windshield wiper).

(1) Is the actuator being used to move a wheeled robot?

Drive motors must move the weight of the entire robot and will most likely require a gear down. Most robots use "skid steering" while cars or trucks tend to use rack-and-pinion steering. If you choose skid steering, DC gear motors are the ideal choice for robots with wheels or tracks as they provide continuous rotation, and can have optional position feedback using optical encoders and are very easy to program and use. If you want to use rack-and-pinion, you will need one drive motor (DC gear is also suggested) and one motor to steer the front wheels). For stirring, since the rotation required is restricted to a specific angle, an R/C servo would be the logical choice.

(2) Is the motor being used to lift or turn a heavy weight?

Lifting a weight requires significantly more power than moving a weight on a flat surface. Speed must be sacrificed in order to gain *torque* and it is best to use a gearbox with a high gear ratio and powerful DC motor or a DC linear actuator. Consider using system (either with worm gears, or clamps) that prevents the mass from falling in case of a power loss.

(3) Is the range of motion limited to 180 degrees?

If the range is limited to 180 degrees and the torque required is not significant, an R/C servo motor is ideal. Servo motors are offered in a variety of different torques and sizes and provide angular position feedback (most use a potentiometer, and some specialized ones use optical encoders). R/C servos are used more and more to create small walking robots.

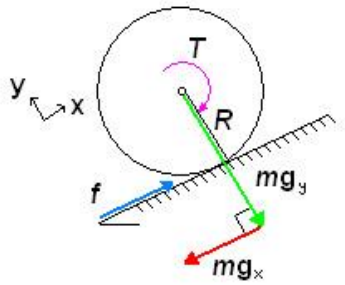
(4) Does the angle need to be very precise?

Stepper motors and geared stepper motors (coupled with a stepper motor controller) can offer very precise angular motion. They are sometimes preferred to servo motors because they offer continuous rotation. However, some high-end digital servo motors use optical encoders and can offer very high precision.

(5) Is the motion in a straight line?

Linear actuators are best for moving objects and positioning them along a straight line. They come in a variety of sizes and configurations. Muscle wire should be considered only if your motion requires very little force. For very fast motion, consider pneumatics or solenoids, and for very high forces, consider DC linear actuators (up to about 500 pounds) and then hydraulics.

Tools



In order to compute the strength (or torque), and speed required for your application, many (rather complex) computations are required involving the physics of the machine to be created. In order to simplify the design process, we have put together a few tools that can help you out.

- DC Drive Motor Selector (useful for robots with wheels or tracks).
  Also consult the Drive Motor Sizing Tutorial for further details.
- Robot Leg Torque Tutorial
- Robot Arm Torque Calculator

Practical Example

In lesson 1 we determined the objective of our project would be to get a better understanding of mobile robots, while keeping the budget to about $200 to a maximum of $300. In lesson 2 we decided we wanted a small tank (on tracks) that could operate on top of a desk.

First, let us determine the type of actuators that would be required by answering the five aforementioned questions:

1. *Is the actuator being used to move a wheeled robot?*
   Yes. A DC gear motor is the suggested type of actuator and skid steering is appropriate for a tank, which means that each track will need it;s own motor.
2. *Is the motor being used to lift or turn a heavy weight?*
   No, a desktop rover should not be heavy.
3. *Is the range of motion limited to 180 degrees?*
   No, the wheels need to urn continuously.
4. *Does the angle need to be precise?*
   No, our robot does not require positional feedback.
5. *Is the motion in a straight line?*
   No, since we want the robot to turn and move in all directions.

Since rotating a wheel needs rotational motion, we could quickly eliminate all linear actuators and choose a DC gear motor. The next logical question was “which one?”A search online shows that there are not too many track systems intended for small robots, which in itself would restrict which motors we could consider.

The Currently Available Track Systems

At 2″ and 3″ wide, the Lynxmotion tracks are more intended for medium sized robots, so we’ll omit them. The price does fall within the budget though.

The Vex Tank Tread Kit is definitely a good option, but it would restrict us to one specific motor.

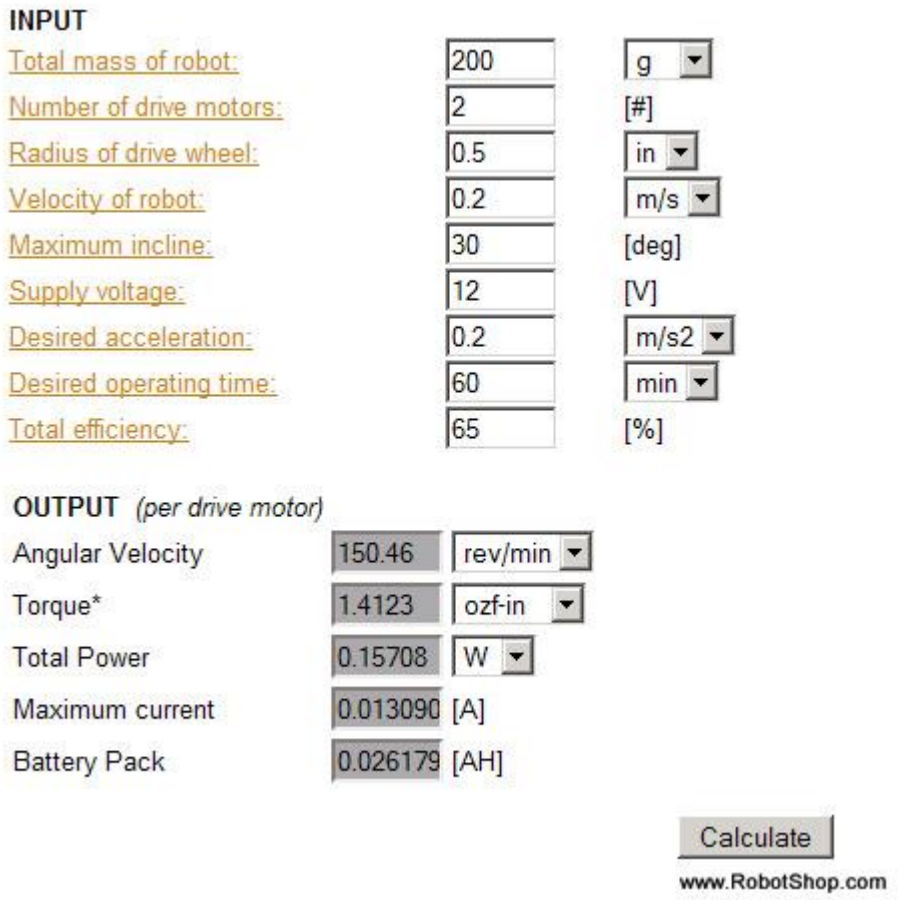
The Tamiya Track and Wheel Set is definitely a good option, and would limit our choices to Tamiya motors and gearboxes. This would also be within the budget.

There are several Johnny Robot Track Kits, one for a Hitec continuous rotation servo (which is essentially a gear motor in a servo’s body) another for a Futaba continuous rotation servo, one for Tamiya motors and another for Pololu or Solarbotics motors. This is definitely a good option and also within our budget. Mainly because of aesthetic and motor compatibility reasons, we are going to stick with this choice.

There is always the option of hacking a toy such as an R/C tank and convert it into a robot. This option would also give us compatible motors, however, the objective is to design our own robot and not hack another product.

Computing the motor requirements

The next step is to fill out the DC Drive Motor Selector Tool, using approximate values.



Data Details

- Total mass of robot:200 g should include everything: motors, frame, batteries and all.
- Number of drive motors:Two motors are required for skid steering.
- Radius of drive wheel: from 0.5" to about 1" should be an appropriate size for a desktop robot.
- Velocity of robot:0.2 m/s would be nice for a desktop robot.
- Maximum incline: Climbing some books would be cool, let us choose 30 degrees.
- Supply Voltage:Uncertain at the moment, so we choose the default 12 V
- Desired Acceleration:Not sure, so choose default 0.2 m/s^2
- Desired operating time: 30 minutes is reasonable between charges.
- Total efficiency:Not sure, so we choose default 65%

Using 0.5 as the wheel radius we obtain 150 rpm @ 1.4 oz-in. When using 1", the calculator provides 75rpm @ 2.8 oz-in.

Selecting the Motor

Thus, the motors we are looking for must turn at approximately 150 rpm and provide roughly 1.4123 oz-in of torque. We can use the DC motor Comparison Table in order to find the appropriate motor.

There are many motors available that fit the Johnny Robot Track Kit :

The Solarbotics GM8 and GM9 feature 70 rpm @ 43 oz-in and 66 rpm at 43 oz-in respectively. Both sell for $5.46 each.

All Tamiya gearbox ad motor combinations sell for approximately $11 and up and provide a wide range of torques and speeds.

Hitec continuous rotation servo and Futaba continuous rotation servos sell for $17 and $14 respectively.

In the end, we opted to use a pair of Solarbotics GM9 in order to use skid-drive, mainly because of their low cost.

It is important to note that although the calculator specified we needed about 150rpm, we chose the motor anyway, knowing it would move at about half the original (desired) velocity. The torque produced by this motor is significantly greater than what we needed, which means it can carry additional weight, or climb stepper angles.

For further information on learning how to make a robot, please visit the RobotShop Learning Center. Visit the RobotShop Community Forum in order to seek assistance in building robots, showcase your projects or simply hang-out with other fellow roboticists.

Tags: Grand Tutorial Series How To Make a Robot Tutorial

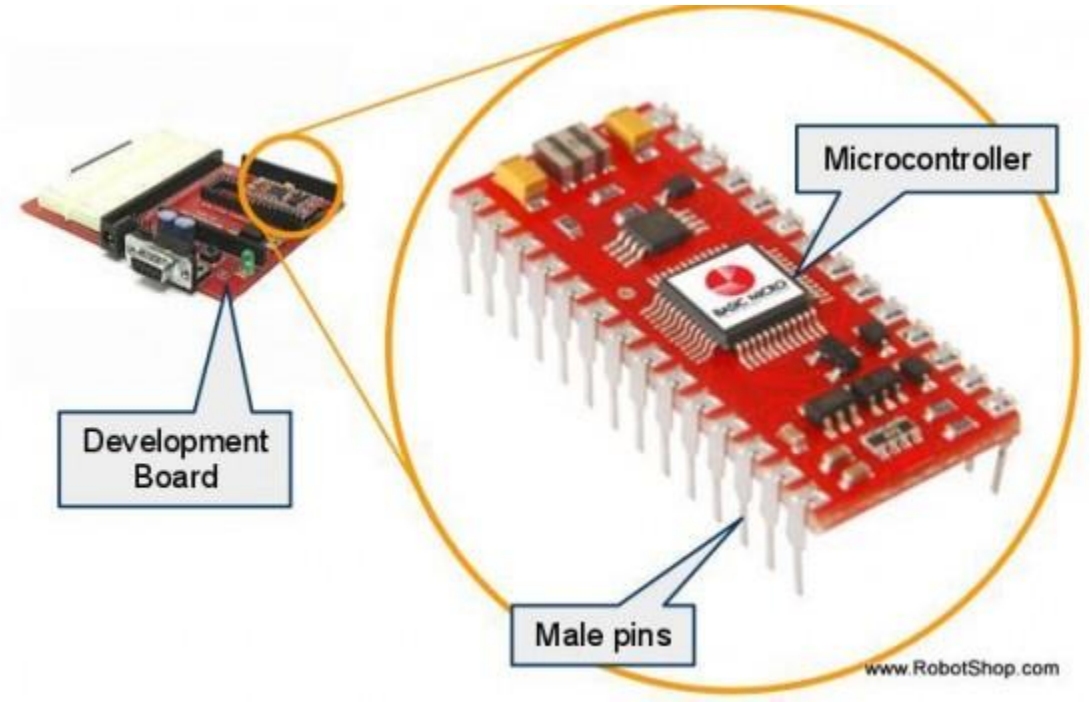
Lesson 4

What is a microcontroller?

You might be asking yourself what is a microcontroller and what does it do?

A microcontroller is a computing device capable of executing a program (i.e. a sequence of instructions) and is often referred to as the “brain” or “control center” in a robot since it is usually responsible for all computations, decision making, and communications.

In order to interact with the outside world, a microcontroller possesses a series of pins (electrical signal connections) that can be turned HIGH (1/ON), or LOW (0/OFF) through programming instructions. These pins can also be used to read electrical signals (coming form sensors or other devices) and tell whether they are HIGH or LOW.



Most modern microcontrollers can also measure analogue voltage signals (i.e. signals that can have a full range of values instead of just two well defined states) through the use of an Analogue to Digital Converter (ADC). By using the ADC, a microcontroller can assign a numerical value to an analogue voltage that is neither HIGH nor LOW.

What can a microcontroller do?

Although microcontrollers can seem rather limited at first glance, many complex actions can be achieved by setting the pins HIGH and LOW in a clever way. Nevertheless, creating very complex algorithms (such as advanced vision processing and intelligent behaviours) or very large programs may be simply impossible for a microcontroller due to its inherent resource and speed limitations.

For instance, in order to blink a light, one could program a repeating sequence where the microcontrollers turns a pin HIGH, waits for a moment, turns it LOW, waits for another moment and starts again. A light connected to the pin in question would then blink indefinitely.

In a similar way, microcontrollers can be used to control other electrical devices such as actuators (when connected to motor controllers), storage devices (such as SD cards), WiFi or Bluetooth interfaces, etc. As a consequence of this incredible versatility, microcontrollers can be found in everyday products. Practically every home appliance or electronic device uses at least one (often many) microcontroller. For instance TV sets, washing machines, remote controls, telephones, watches, microwave ovens, and now robots require these little devices to operate.

Unlike microprocessors (e.g. the CPU in personal computers), a microcontroller does not require peripherals such as external RAM or external storage devices to operate. This means that although microcontrollers can be less powerful than their PC counterpart, developing circuits and products based on microcontrollers is much simpler and less expensive since very few additional hardware components are required.

It is important to note that a microcontroller can output only a very small amount of electrical power through its pins; this means that a generic microcontroller will likely not be able to power electrical motors, solenoids, large lights, or any other large load directly. Trying to do so may even cause physical damage to the controller.

What are the more specialized features in a microcontroller?

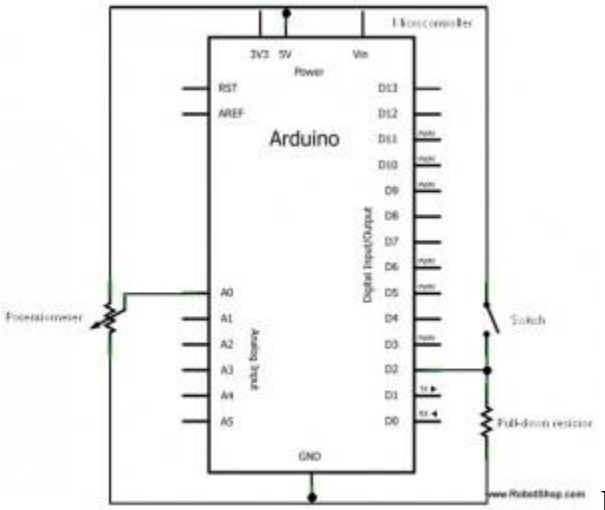
Special hardware built into the microcontrollers means these devices can do more than the usual digital I/O, basic computations, basic mathematics, and decision taking. Many microcontrollers readily support the most popular communication protocols such as UART (a.k.a. serial or RS232), SPI and I^2C.This feature is incredibly useful when communicating with other devices such as computers, advanced sensors, or other microcontrollers. Although it is possible to manually implement these protocols, it is always nice to have dedicated hardware built-in that takes care of the details. It allows the microcontroller to focus on other tasks and allows for a cleaner program.

Analogue-to-digital converters (ADC) are used to translate analogue voltage signals to a digital number proportional to the magnitude of the voltage, this number can then be used in the microcontroller program. In order to output an intermediate amount of power different from HIGH and LOW, some microcontrollers are able to use pulse-width modulation (PWM). For example this method makes it possible to smoothly dim an LED.

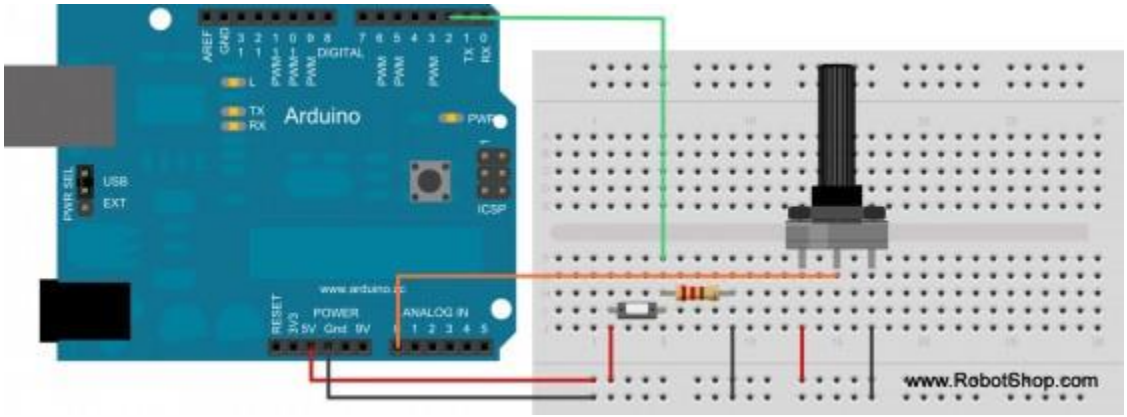
Finally, some microcontrollers integrate a voltage regulator in their development boards. This is rather convenient since it allows the microcontroller to be powered by a wide range of voltages that do not require you to provide the exact operating voltage required. This also allows it to readily power sensors and other accessories without requiring an external regulated power source.

Analogue or Digital?

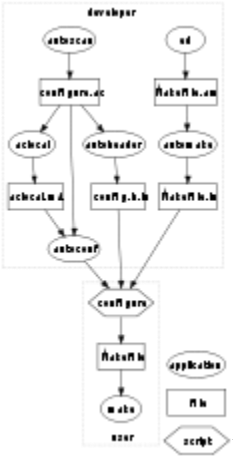
Below you can find two examples that illustrate when to use a digital or analogue pin:

1. 

Digital: A digital signal is used in order to assess the binary state of a switch. As illustrated below (on the left side of the solderless breadboard), a momentary switch or push button closes a circuit when pressed, and allows current to flow (a pull-up resister is also shown). A digital pin connected (through a green wire in the picture) to this circuit would return either LOW or 0 (meaning that the voltage at the pin is in the LOW range, 0V in this case) or a HIGH (meaning the button is pressed and the voltage is at the HIGH range, 5V in this case).
2. **Analogue:** A variable resistor or potentiometer (as shown towards the right side of the board below) is used to provide an analogue electrical signal proportional to a rotation (e.g. the volume knob on a stereo). As illustrated below, when a potentiometer is connected to a 5V supply and the shaft is turned, the output will vary between 0 and 5V, proportionally to the angle of rotation. The ADC on a microcontroller interprets the voltage and converts it to a numeric value. For example, a 10-bit ADC converts 0V to the value “0”, 2.5V to “512” and 5V to “1023”. Therefore if you suspect the device you plan to connect will provide a value that is proportional to something else (for example temperature, force, position), it will likely need an analogue pin.



What about programming?



Being afraid of programming microcontrollers is getting old fashioned. Unlike the “old days” where making a light blink took advanced knowledge of the microcontroller and several dozen lines of code (not to mention parallel or serial cables connected to huge development board), programing a microcontroller is very simple thanks to modern Integrated Development Environments (IDE) that use up-to-date languages, fully featured libraries that readily cover all of the most common (and not so common) action, and several ready-made code examples to get beginners started.

Now-a-days, microcontrollers can be programmed in various high-level languages including C, C++, C#, Processing (a variation of C++), Java, Python, .Net, and Basic. Of course, it is always possible to program them in Assembler but this privilege is reserved for more advanced users with very special requirements (and a hint of masochism). In this sense, anyone should be able to find a programming language that best suit their taste and previous programming experience.

IDEs are becoming even simpler as manufacturers create graphical programming environments. Sequences which used to require several lines of code are reduced to an image which can be connected to other “images” to form code. For example, one image might represent controlling a motor and the user need only place it where he/she wants it and specify the direction and rpm.

On the hardware side, microcontroller developments boards add convenience and are easier to use over time. These boards usually break out all the useful pins of the microcontroller and make them easy to access for quick circuit prototyping. They also provide convenient USB power and programming interfaces that plug right into any modern computer.

For those unfamiliar with the term, a Development Board is a circuit board that provides a microcontroller chip with all the required supporting electronics (such as voltage regulator , oscillators, current limiting resistors, and USB plugs) required to operate. If you are not planning to design your own support circuit, buying a development board is preferable to simply getting a single microcontroller chip.

Note: Robot programming is covered in greater depth in Lesson 10.

Why not use a standard computer?

It is apparent that a microcontroller is very similar to a PC CPU or microprocessor, and that a development board is akin to a Computer motherboard. If this is the case, why not simply use a full computer to control a robot?

As a matter of fact, in more advanced robots, especially those that involve complex computing and vision algorithms, the microcontroller is often replaced (or supplemented) with a standard computer. A desktop computer includes a motherboard, a processor, a main storage device (such as a hard drive), video processing (on-board or external), RAM, and of course peripherals such as monitor, keyboard, mouse etc. This type of system is usually more expensive, physically larger, more power hungry. The main differences are highlighted in the table below.

	Microcontroller	**Personal Computer**
Example	Atmega328	Intel Pentium Core 2 Duo
RAM	1KB	4000000KB (4GB)
Storage	15KB	15000000KB (1000GB)
Power	0.1W	600W
Voltage	12	12
Input/Output	Pins	USB, RS232
Wireless	Bluetooth*, RF*	Bluetooth
Video	None	1000000KB (1GB)
Price	$4 to $300	$400 to $2000

Internet WiFi* or Ethernet*WiFi or Ethernet
*Available as optional additions on many microcontrollers.

Choosing the right Microcontroller

Unless you are into BEAM robotics, or plan to control your custom robot using a tether or an R/C system (which, based on our definition from Lesson 1 would not be considered a robot), you will need a microcontroller for any robotic project. For a beginner, choosing the right microcontroller may seem like a daunting task, especially considering the range of products, specifications and potential applications. There are many different microcontrollers available on the market: Arduino, BasicATOM, BasicX, POB Technology, Pololu, Parallax and more. When considering the right microcontroller, ask yourself the following questions:

1. **Which microcontroller is the most popular for my application?**
   Of course making robots or electronic projects in general is not a popularity contest, but the fact that a microcontroller has a large supporting community or has been successfully used in a similar (or even the same) situation could simplify your design phase considerably. This way, you could benefit from other user's experience and among hobbyists. It is common for robot builders to share results, code, pictures, videos, and detail successes and even failures. All this available material and the possibility of receiving advice from more experienced users can prove very valuable.
2. **Does it have any special features the robot requires?**
   As popular as a microcontroller might be, it must be able to perform all the special actions required for your robot to functions properly. Some features are common to all microcontrollers (e.g. having digital inputs and outputs, being able to perform simple mathematical operations, comparing values and taking decisions), while others need specific hardware (e.g ADC, PWM, and communication protocol support). Also memory and speed requirements, as well as pin count should be taken into consideration.
3. **Are the accessories I need available for a particular microcontroller?**
   If your robot has special requirements or there is a particular accessory or component that is crucial for your design, choosing a compatible microcontroller is obviously very important. Although most sensors and accessories can be interfaced directly with many microcontrollers, some accessories are meant to interface with a specific microcontroller and even provide out-of-the-box functionally or sample code.

What does the future hold?

As the price of computers has gone down, and advances in technology make them smaller and more energy efficient, single-board computer have emerged as an attractive option for robots. These single-board computers are essentially computers you may have used about 5 years ago, and incorporate many devices into one board (so you cannot swap anything out). They can run a complete operating system (Windows and Linux are most common) and can connect to external devices such as USB peripherals, LCDs etc. Unlike their ancestors, these single-board computers tend to be much more power efficient.

Practical Example

In order to choose a microcontroller, we compiled a list of features / criteria we wanted:

1. The microcontroller's cost must be low while including a development board (below 50$)
2. It must be easy to use and well supported. It is also important to have lots of documentation readily available.
3. It should be programmed in C or a C-based language.
4. It must be popular and have an active user community.
5. Since the robot will be used as a general purpose platform, the microcontroller should be very feature rich in order to allow for broad experimentation. In this sense, it should have several analogue and digital pins, as well as an integrated voltage regulator.

Since our robot will use two motors, the microcontroller will need two digital pins for direction control, and two PWM pins for speed control (this will be explained in more detail in Lesson 5). The robot will also transmit and receive data so it will need to support the UART (a.k.a. serial or RS232) communication protocol in our case. We would also like the option of adding other sensors and devices in the future so analogue pins and many extra digital pins would be appropriate.

The upcoming RobotShop Microcontroller comparison table allows us to compare the main features of one microcontroller with another. The Pololu and Arduino microcontrollers seemed to conform best to the above criteria. In order to select a specific microcontroller from these two manufacturers, each was researched in order to determine the amount of available material, code, user community, Google hits and more.

The Arduino Duemilanove (recently replaced by the Arduino Uno) was ultimately chosen based on price vs. features and because of the concept of "shields" (separate accessory boards you plug and stack onto the microcontroller which add specific functionality). Also, Arduino is rather popular, there are many sample projects, and its community is very active.

For further information on learning how to make a robot, please visit the RobotShop Learning Center. Visit the RobotShop Community Forum in order to seek assistance in building robots, showcase your projects or simply hang-out with other fellow roboticists.

Tags: Arduino Tutorials Grand Tutorial Series How To Microcontroller

Lesson 5

Lessons Menu:

Now that the general shape, the actuators (or motors) and the brain for the robot have been chosen, it is time to make things move.

The first question many beginners have when building their first robot is "*how do I control the motors?*" After a bit of research, the word *motor controller* comes up a lot.

What is a motor controller and why do I need it?

A motor controller is an electronic device (usually comes in the shape of a bare circuit board without enclosure) that acts as an intermediate device between a microcontroller, a power supply or batteries, and the motors.

Although the microcontroller (the robot's brain) decides the speed and direction of the motors, it cannot drive them directly because of its very limited power (current and voltage) output. The motor controller, on the other hand, can provide the current at the required voltage but cannot decide how fast the motor should turn.

Thus, the microcontroller and the motor controller have to work together in order to make the motors move appropriately. Usually, the microcontroller can instruct the motor controller on how to power the motors via a standard and simple communication method such as UART (a.k.a. serial) or PWM. Also, some motor controllers can be manually controlled by an analogue voltage (usually created with a potentiometer).

The physical size and weight of a motor controller can vary significantly, from a device smaller than the tip of your finger used to control a mini sumo robot to a large controller weighing several Kg. The weight and size of a motor controller usually has a minimal impact on the robot, until you get into small robotics or unmanned aerial vehicles. The size of a motor controller is usually related to the maximum current it can provide. Larger current also means having to use larger diameter wires (the smaller the gauge number, the larger the diameter).

Motor Controller Types

Since there are several types of actuators (as discussed in lesson 3), there are several types of motor controllers:

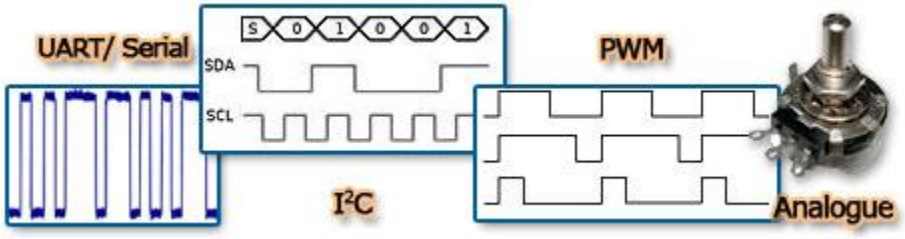
- Brushed DC motor controllers: used with brushed DC, DC gear motors, and many linear actuators.
- Brushless DC motor controllers: used with brushless DC motors.
- Servo Motor Controllers: used for hobby servo motors
- Stepper Motor Controllers: used with unipolar or bipolar stepper motors depending on their kind.

Choosing a Motor Controller

Motor controllers can only be chosen after you have selected your motors/actuators. Also, the current a motor draws is related to the torque it can provide: a small DC motor will not consume much current, but cannot provide much torque, whereas a large motor can provide higher torque but will require a higher current to do so.

DC Motor Control:

1. The first consideration is the motor's **nominal voltage**. DC motor controllers tend to offer a voltage range. For example, if your motor operates at 3V nominal, you should not select a motor controller that can only control a motor between 6V and 9V. This will help you cross off some motor controllers from the list.
2. Once you have found a range of controllers that can power the motor with the appropriate voltage, the next consideration is the **continuous current**the controller will need to supply.You need to find a motor controller that will provide current equal to or above the motor's continuous current consumption under load. Should you choose a 5A motor controller for a 3A motor, the motors will only take as much current as they require. On the other hand, a 5A motors is likely to burn a 3A motor controller.Many motor manufacturers provide a DC motor's stall current, which does not give you a clear idea of the motor controller you will need. If you cannot find the motor's continuous operating current, a simple rule of thumb is to estimate the motor's continuous current at about 20% to 25% of the stall current. All DC motor controllers provide a **maximum** current rating – be certain this rating is about double that of the motor's continuous operating current. Note that when a motor needs to produce more torque (for example going up an incline), it requires more current. Choosing a motor controller with built-in over current and thermal protection is a very good choice.
3. The **Control method** is another important consideration. Control methods include analogue voltage, I^2C, PWM, R/C, UART (a.k.a. serial). If you are using a microcontroller, check to see which pin types you have available and which motors are viable for you to choose. If your microcontroller has serial communication pins, you can choose a serial motor controller; for PWM, you will likely need one PWM channel per motor.



4. The final consideration is a practical one: **Single vs. dual** (double) motor controller. A dual DC motor controller can control the speed and direction of two DC motors independently and often saves you money (and time). The motors do not need to be identical, though for a mobile robot, the drive motors should be identical in most cases. You need to choose the dual motor controller based on the more powerful DC motor. Note that dual motor controllers tend to have only one power input, so if you want to control one motor at 6V and the other at 12V, it will not be possible. Note that the current rating provided is almost always **per channel**.

Servo Motor Control:

Since standard hobbyist servo motors are meant to use specific voltages (for peak efficiency), most operate at 4.8V to 6V, and their current consumption is similar, the steps for the selection are somewhat simplified. However, you may find a servo motor that operates at 12V; it is important to do additional research about a servo motor controller if your servo motor is not considered "standard".

Also, most hobby servo motors use the standard R/C servo input (three wires which are **ground**, **voltage** and **signal**)

1. Choose the **control method**. Some servo motor controllers allow you to control the servo's position manually using a dial/switch/buttons, while others communicate using UART (serial) commands or other means.
2. Determine the number of servos to be controlled . Servo controllers can control many servos (usually 8, 16, 32, 64 and up). You can certainly select a servo motor controller capable of controlling more servos than you will need.
3. As with DC motor controllers, the control method is an important consideration.

Stepper Motor Control:

1.  Is the motor you selected **unipolar or bipolar**? Choose a stepper motor controller type accordingly, though a growing number are able to control both types. The number of leads is usually a dead give-away of the motor type: if the motor has 4 leads, then it is bipolar; should it have 6 or more leads, then it is unipolar.
2. Choose the motor controller voltage range to match your motor's **nominal voltage** .
3. Find out how much **current per coil**your motor requires, and find out how much current (per coil) the stepper motor controller can provide.If you cannot find the current per coil, most manufacturers list the coil impedance, R . Using Ohms Law (V=IR), you can then calculate the current (I).
4. As with DC motor controllers, the control method is an important consideration.

Linear Actuator Control:

Linear actuators come in three main flavours regarding their control method.: DC, R/C, or position feedback.

Most DC linear actuators use a geared DC motor, so a DC motor controller is usually appropriate. However, some linear actuators take R/C servo input, so you would select a servo motor controller. Should an R/C controlled linear actuator operate at a higher voltage than the servo controller's range, the actuator may include separate wires for the higher supply voltage required.

Other Actuators:

Many "miscellaneous" electromechanical devices such as muscle-wire, solenoids, or even powerful lights need to be controlled using motor controllers. Below are some questions to determine if your actuator might need a motor controller:

- Higher current requirements: any device that requires over 0.1A usually needs its own controller
- Higher voltage requirements: if the actuator operates above the microcontroller's voltage (usually 5V or 3.3V), it usually cannot be directly connected to a microcontroller

For more information regarding actuator control and communications method, please visit the RobotShop Learning Center.

Practical Example

In the previous lesson, we had chosen the Solarbotics GM9 gear motors.

Below are this motor's specifications:

- Gear Ratio: 143:1
- Unloaded RPM (3V): 40
- Unloaded RPM (6V): 78
- Unloaded Current (3V): 50mA
- Unloaded Current (6V): 52mA
- Stall Current (3V): 400mA
- Stall Current (6V): 700mA
- Stall Torque (3V) : 44.44in*oz
- Stall Torque (6V) : 76.38in*oz

Applying the steps:

1. The nominal voltage is 3V or 6V.
2. There is no mention of continuous current, though the stall torque at both voltages is provided: 400mA and 700mA. If we take 25% of these values, the continuous current can be approximated at 100mA to 175mA. To be safe we can take the larger value.
3. We have chosen a microcontroller that has many different pins including serial, PWM, analog and digital.
4. Our little rover will be using two identical motors, so we can use a dual motor controller.

Given the above criteria, we are looking for a motor controller with the following specifications:

- Voltage range can accommodate a 3V to 6V motor
- Continuous current at least 350mA per channel (low power category)
- Communication method is PWM, I2C or analog (or several of these)
- Dual motor control is preferred.

By Looking at the Brushed DC Motor Controllers Comparison Table (imperial version), several motor controllers fit the criteria:

- RB-Dim-19 (6-18V, 5A, dual. Analogue and Serial interfaces with many safety features)
- RB-Pol-16 (1.5-6V, 5A, dual. Low cost controller with serial interface)
- RB-Pol-22 (6-16V, 9A, dual, PWM interface)
- RB-Spa-397 (5-16V, 2A, dual, serial interface)
- RB-Ada-02 (4.5-36V, 0.6A, dual. Arduino shield with PWM interface)
- RB-Cri-15 (6-58V, 10A, single, PWM)
- RB-Cri-14 (6-58V, 10A, single, PWM)
- … and many more.

There are a variety of other motor controllers which meet the criteria above which would work as well. In order to reduce this list, cost and features would need to be considered. For example, there is no need to consider a high current (10A) motor controller which is understandably more expensive than a 5A controller. We can also eliminate all single motor controllers. The one controller that stands out from the rest is RB-Pol-16 because of its lower voltage range; this means that, should we decide to power the motor at 3V, it would fall within this controller's voltage range. The other controller of interest is RB-Ada-02 because it is made specifically for

the microcontroller we selected (i.e the Arduino Uno). However, the one downside to RB-Ada-02 is that no additional shields can be installed afterwards. The Pololu dual motor controller was ultimately chosen because of its lower voltage range and price.

For further information on learning how to make a robot, please visit the RobotShop Learning Center. Visit the RobotShop Community Forum in order to seek assistance in building robots, showcase your projects or simply hang-out with other fellow roboticists.

Tags: actuator Grand Tutorial Series How To Motor Controllers

Lesson 6

Lessons Menu:

The definition we have chosen for a "robot" requires the device to obtain data about its environment, make a decision, and then take action accordingly. This does not exclude the option of a robot being semi-autonomous (having aspects which are controlled by a human and others which it does on its own). A good example of this is a sophisticated underwater robot; a human controls the basic movements of the robot while an on-board processor measures and reacts to underwater currents in order to keep the robot in the same position without drifting. A camera onboard the robot sends video back to the human while onboard sensors may track the water temperature, pressure and more. If the robot loses communication with the surface, an autonomous program may kick-in causing it to surface. If you want to be able to send and/or receive commands from your robot, you will need to determine its level of autonomy and if you want it to be tethered, wireless or fully autonomous.

Tethered

Direct Wired Control

The easiest way to control a vehicle is with a handheld controller physically connected to the vehicle using a cable (i.e. a tether). Toggle switches, knobs, levers, joysticks and buttons on this controller allow the user to control the vehicle without the need to incorporate complex electronics. In this situation, the motors and a power source can be connected directly with a switch in order to control its forward/backwards rotation. Such vehicles

usually have no intelligence and are considered to be more “remote controlled machines” than “robots”.

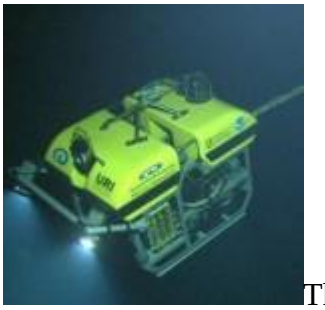
Advantages

- The robot is not limited to an operating time since it can be connected directly to the mains
- There is no worry about loss of signal
- Minimal electronics and minimal complexity
- The robot itself can be light weight or have added payload capacity
- The robot can be physically retrieved if something goes wrong (very important for underwater robots)

Disadvantages

- The tether can get caught or snagged (and potentially cut)
- Distance is limited by the length of the tether
- Dragging a long tether adds friction and can slow or even stop the robot from moving

Wired Computer Control

 The next step is to incorporate a microcontroller into the vehicle but continue to use a tether. Connecting the microcontroller to one of your computer’s I/O ports (e.g. a USB port) allows you to control its actions using a keyboard (or keypad), joystick or other peripheral device. Adding a microcontroller to a project also may require you to program how the robot reacts to the input. Instead of using a laptop or desktop computer, netbooks are often a desirable choice because of their low price, small size and low weight.

Advantages

- Same advantages as with direct wired control
- More complex behaviours can be programmed or mapped to single buttons or commands.
- Larger controller choice (mouse, keyboard, joystick, etc.)
- Added onboard intelligence means it can interface with sensors and make certain decisions on its own

Disadvantages

- Cost is higher than a purely tethered robot because of the added electronics
- Same disadvantages as with direct wired control

Ethernet


A variation on computer control would be to use an Ethernet interface. A robot that is physically connected to a router (so it could be controlled via the Internet) is also possible (though not very practical) for mobile robots. Setting-up a robot that can communicate using the internet can be fairly complex, and more often than not, a WiFi (wireless internet) connection is preferable. A wired and wireless combination is also an option, where there is a transceiver (transmit and receive) connected physically to the internet and data received via the internet is then sent wirelessly to the robot.

Advantages

- Robot can be controlled trough the Internet from anywhere in the world
- The robot is not limited to an operating time since it could use Power over Ethernet (PoE).
- Using Internet Protocol (IP) can simplify and improve the communication scheme.
- Same advantages as with direct wired computer control

Disadvantages

- Programming involved is more complex
- The tether can get caught or snagged (and potentially cut)
- Distance is limited by the length of the tether
- Dragging a long tether adds friction and can slow or even stop the robot from moving

Wireless

Infrared


Infrared transmitters and receivers cut the cables connecting the robot to the operator. This is usually a milestone for beginners. Infrared control requires “line of sight” in

order to function; the receiver must be able to “see” the transmitter at all times in order to receive data. Infrared remote controls (such as universal remote controls for televisions) are used to send commands to an infrared receiver connected to a microcontroller which then interprets these signals and controls the robot’s actions.

Advantages

- Low cost
- Simple TV remote controls can be used as controllers

Disadvantages

- Needs to be line of sight
- Distance is limited

Radio Frequency (RF)

Commercially available Remote Control (R/C) units use small microcontrollers in the transmitter and receiver to send, receive and interpret data sent via radio frequency (RF). The receiver box has a PCB (printed circuit board) which comprises the receiving unit and a small servo motor controller. RF communication requires either a transmitter matched/paired with a receiver, or a transceiver (which can both send and receive data). RF does not require line of sight and can also offer significant range (transmission distance). Standard radio frequency devices can allow for data transfer between devices as far away as several kilometres and there is seemingly no limit to the range for more professional RF units.

XBee and Zigbee modules use RF for communication, but allow the user to vary many of the communication parameters involved. These modules have a specific footprint (layout) and are only produced by certain companies. Their main advantage is that they provide a very robust easy to set up link and take care of all of the communication protocol details.

Many robot builders choose to make semi-autonomous robots with RF capability since it allows the robot to be as autonomous as possible, provide feedback to a user and still give the user some control over some of its functions should the need arise.

Advantages

- Considerable distances possible
- Setup can be straightforward
- Omni directional (impeded but not entirely blocked by walls and obstructions)

Disadvantages

- Very low data rate (simple commands only)
- Pay attention to the transmission frequencies – they can be shared

Bluetooth

Bluetooth is a form of RF and follows specific protocols for sending and receiving data. Normal Bluetooth range is often limited to about 10m though it does have the advantage of allowing users to control their robot via Bluetooth-enabled devices such as cell-phones, PDAs and laptops (though custom programming may be required to create an interface). Just like RF, Bluetooth offers two-way communication.

Advantages

- Controllable from any Bluetooth enabled device (usually additional programming is necessary) such as a Smartphone, laptop, desktop etc.
- Higher data rates possible
- Omnidirectional (does not need line of sight and can travel a little through walls)

Disadvantages

- Devices need to be "paired"
- Distance is usually about 10m (without obstructions)

WiFi

WiFi is now an option for robots; being able to control a robot wirelessly via the internet presents some significant advantages (and some drawbacks) to wireless control. In order to set up a WiFi robot, you need a wireless router connected to the internet and a WiFi unit on the robot itself. For the robot, you can also use a device that is TCP/IP enabled with a wireless router.

Advantages

- Controllable from anywhere in the world so long as it is within range of a wireless router
- High data rates possible

Disadvantages

- Added programming required
- Maximum range is usually determined by the choice of wireless router

GPRS / Cellular

 Another wireless technology that was originally developed for human to human communication, the cell phone, is now being used to control robots. Since cellular frequencies are regulated, incorporating a cellular module on a robot usually requires added patience for programming as well as an understanding of the cellular network system and the regulations.

Advantages

- Robot can be controlled anywhere it has a cellular signal
- Direct satellite connection is possible

Disadvantages

- Setup and configuration can be complex – NOT for beginners
- Each network has its own requirements / restrictions
- Cellular service is not free; usually the more data you transmit/receive the more money you will need to pay.
- System is not (yet) well setup for robotics use

Autonomous

The next step is to use the microcontroller in your robot to its full potential and program it to react to input from its sensors. Autonomous control can come in various forms: pre-programmed with no feedback from the environment, limited sensor feedback and finally complex sensor feedback. True “autonomous control” involves a variety of sensors and code to allow the robot to determine by itself the best action to be taken in any given situation.

The most complex methods of control currently implemented on autonomous robots are visual and auditory commands. For visual control, a robot looks to a human or an object in order to get its commands. Getting a robot to turn to the left by showing a piece of paper with arrow pointing left is a lot harder to accomplish than one might initially suspect. An auditory command such as “turn left” also requires quite a bit of programming. Programming a variety of complex commands like “get me a drink from the fridge” or “get my shoes, they’re near the front door” is no longer fantasy but requires a very high level of programming, and a lot of time.

Advantages

- This is “real” robotics
- Tasks can be as simple as blinking a light based on one sensor readings to landing a spacecraft on a distant planet.

Disadvantages

- It’s only as good as the programmer; if it’s doing something you don’t want it to do, the only option you have is to check your code, modify it and upload the changes to the robot.

Practical Example

For our project, the goal is to create an autonomous rover capable of making a decision based on external input from sensors. Should the robot “misbehave” it will be physically close and shutting it off will not be an issue. However having the option of semi-autonomous (wireless) control to allow us the option of making a remote-controlled vehicle is also attractive. We will not have the need for tethered control.

The microcontroller chosen in the previous lesson uses what are called "shields" which are essentially ad-on boards specific to the Arduino's pin layout. There are many shields, including ones that allow for Ethernet, Xbee, or Bluetooth communication. There is even a shield that allows for GPRS (i.e. cellular) communications. The basic robot will therefore have no additional modules, though it is important to note that it does have wireless communication capability.

For further information on learning how to make a robot, please visit the RobotShop Learning Center. Visit the RobotShop Community Forum in order to seek assistance in building robots, showcase your projects or simply hang-out with other fellow roboticists.

Tags: Grand Tutorial Series How To Make a Robot

Make a Robot - RobotShop Blog
Posts about Make a Robot - RobotShop Blog
Read more...

Remove preview

Lesson 7

Lessons Menu:

- Lesson 8 – Getting the Right Tools
- Lesson 9 – Assembling a Robot
- Lesson 10 – Programming a Robot

Unlike humans, robots are not limited to just sight, sound, touch, smell and taste. Robots use a variety of different electromechanical sensors to explore and understand their environment and themselves. Emulating a living creature's senses is currently very difficult, so researchers and developers have resorted to alternatives to biological senses.

What can humans sense that robots can't?

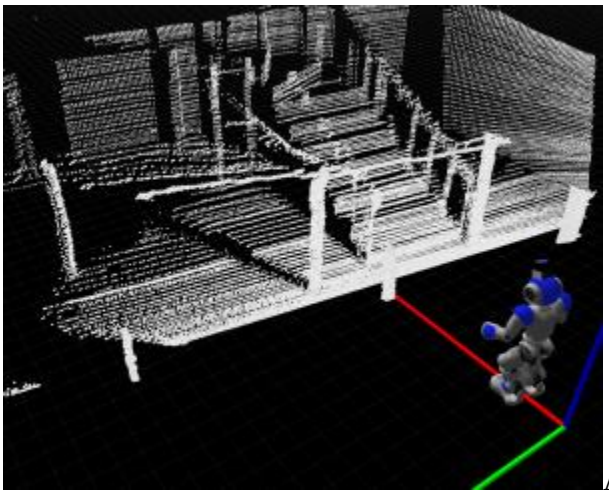
Robots can "see" but have a hard time understanding what they are looking at. Using a camera, a robot may be able to pick up an image made up of millions of pixels but without significant programming, it would not know what any of those pixels meant. Distance sensors would indicate the distance to an object, but would not stop a robot from bumping into it. Researchers and companies are experimenting with a variety of different approaches to permit a robot to not only "see" but "understand" what it is looking at. It may be a long time before a robot is able to differentiate between objects placed before it on a table, especially if they do not appear to be exactly the same as what is in its database of objects.

Robots have a really hard time tasting and smelling. A human may be able to tell you "this tastes sweet" or "this smells bad" whereas a robot would need to analyze the chemical composition and then look up the substance in a database to determine if humans have marked the taste as being "sweet" or the smell as being "bad". There has not been much demand for a robot that can taste or smell, so not much effort has been put into creating the appropriate sensors.

Humans have nerve endings throughout their skin and as such, we know when we have touched an object or when something has touched us. Robots are equipped with buttons or simple contacts placed in strategic locations (for example on a front bumper) to determine if it has come into contact with an object. Robot pets may have contact or force sensors placed in their head,

feet and back, but if you try to touch an area where there is no sensor, the robot has no way of knowing it has been touched and will not react. As research into humanoid robots continues, perhaps an “electromechanical skin” will be developed.

What can robots sense that humans can’t?



Although a robot cannot tell you if a substance tastes good or if an odour smells bad, the steps involved in analyzing the chemical composition can give it far more information than a normal human could about its properties. A robot, equipped with a carbon monoxide sensor, would be able to detect carbon monoxide gas which is otherwise colorless, odorless to humans. A robot would also be able to tell you the Ph level of a substance to determine if it is acidic or basic and much, much more.

Humans use a pair of eyes to get a very good sense of depth, though for many, accurately gauging distance is not easy. A human might tell you “the tree looks to be about 50 feet away”, but a robot, equipped with the right distance sensors, can tell you “the tree is 43.1 feet away”.

Additionally, robots can not only sense but give accurate values of a variety of environmental factors that humans are otherwise unaware of or incapable of sensing. For example, a robot can tell you the precise angular or linear acceleration it is subjected to, while most humans would only tell you “I’m turning”, or “I’m moving”. A human can tell you based on experience if they think an object will be hot or cold without actually touching it, whereas a thermal camera can provide a 2D thermal image of whatever is in front of it. Although humans have five main senses, robots can have an almost infinite number of different sensors.

Which sensors do my robots need?

So, what types of sensors are available and which ones does your robot need? You need to first ask yourself “what do I want or need the robot to measure?” and refer to the appropriate category below. There is a good chance what you have in mind will not fall “nicely” into one of these categories, so try to break it down into its basic elements.

Contact

Push button / Contact switch

Switches, buttons, and contact sensors are used to detect physical contact between objects and are not just restricted to humans pushing buttons; bumpers on a robot can be equipped with momentary push buttons, and “whiskers” (just like an animal) can be used to sense multiple distances.

- Advantages: very low cost, easy to integrate, reliable
- Disadvantages: single distance measurement

Pressure sensor

Unlike a push button which offers one of two possible readings (ON or OFF), a pressure sensor produces an output proportional to the force that is being applied to it.

- Advantages: allows gauging how much force is being applied
- Disadvantages: can be imprecise and are more difficult to use than simple switches.

Distance



Ultrasonic Range Finders

Ultrasonic range finders use acoustics to measure the time between when a signal is sent versus when its echo is received back. Ultrasonic range finders can measure a range of distances, but are used specifically in air and are affected by the reflectivity of different materials.

- Advantages: medium range (several meters) measurement.
- Disadvantages: surfaces and environmental factors can affect the readings.

Infrared

Infrared light, which as we saw is used in communication, can also be used to measure distance. Some infrared sensors measure one specific distance while others provide an output proportional to the distance to an object.

- Advantages: low cost, fairly reliable and accurate.
- Disadvantages: closer range than ultrasonic

Laser

Lasers are used when high accuracy, or long distances (or both) are required when measuring the range to an object. Scanning laser rangefinders use spinning lasers to get a two dimensional scan of the distances to objects

- Advantages: very accurate, very long range.
- Disadvantages: much costlier than regular infrared or ultrasonic sensors.

Encoders

Optical encoders use mini infrared transmitter/receiver pairs and send signals when the infrared beam is broken by a specifically designed spinning disk (mounted to a rotating shaft). The number of times the beam is broken corresponds to the total angle travelled by a wheel. Knowing the radius of the wheel, you can determine the total distance travelled by that wheel. Two encoders give you a relative distance in two dimensions.

- Advantages: assuming there is no slip, the displacement is absolute. Often comes installed on the rear shaft of a motor
- Disadvantages: additional programming required; more accurate optical encoders can be ~$50+ each

Linear Potentiometer, resistive band

A linear potentiometer is able to measure the absolute position of an object. A resistive band changes resistance depending on where a force is applied.

- Advantages: position is absolute. A resistive band requires pressure to be applied at a given position.
- Disadvantages: range is very small

Stretch and Bend Sensors

A stretch sensor is made up of a material whose resistance changes according to how much it has been stretched. A bend sensor is usually a sandwich of materials where the resistance of one of the layers changes according to how much it has been bent. These can be used to determine a small angle or rotation, for example how much a finger has been bent.

- Advantages: useful where an axis of rotation is internal or inaccessible
- Disadvantages: not very accurate, and only small angles can be measured

Stereo Camera System

Just like human eyes, two cameras placed a distance apart can provide depth information (stereo vision). Robots equipped with cameras can be some of the most capable and complex robots produced. A camera, combined with the right software, can provide color and object recognition.

- Advantages: can provide dept information and a good feedback about a robot's environment
- Disadvantages: complex to program and use the information

Positioning



Indoor Localization (room navigation)

An indoor localization system can use several beacons to triangulate the robot's position within a room, while others use a camera and landmarks.

- Advantages: excellent for absolute positioning
- Disadvantages: requires complex programming and the use of markers

GPS

A GPS uses the signals from several satellites orbiting the planet to help determine its geographic coordinates. Regular GPS units can provide geographical positioning down to 5m of accuracy while more advanced systems involving data processing and error correction thanks to the use of other GPS units or IMUs can be accurate down to several cm.

- Advantages: does not requires markers or other references
- Disadvantages: can only function outdoors.

Rotation

Potentiometer

A rotary potentiometer is essentially a voltage divider and provides an analog voltage corresponding to the angle the knob is rotated to.

- Advantages: simple to use, inexpensive, reasonably accurate, provides absolute readings.
- Disadvantages: most are restricted to 300 degrees of rotation

Gyroscope

An electronic gyroscope measures the rate of angular acceleration and provides a corresponding signal (analog voltage, serial communication, I2C etc.). Integrating this value twice will give you an angle.

- Advantages: no moving “mechanical” components
- Disadvantages: the sensor is always subjected to angular acceleration whereas a microcontroller cannot always take continuous input, meaning values are lost, leading to “drift”.

Encoders

Optical encoders, as explained above, use mini infrared transmitter/receiver pairs to signal when the infrared beam is broken by a spinning disk (mounted to a rotating shaft). The number of times the beam is broken corresponds to the total angle travelled by a wheel. A mechanical encoder uses a very finely machined disk with enough holes to be able to read specific angles. Mechanical encoders can therefore be used for both absolute and relative rotation.

- Advantages: accurate
- Disadvantages: for optical encoders, the angle is relative (not absolute) to the starting position.

Environmental Conditions

Light Sensor

A light sensor can be used to measure the intensity of a light source, be it natural or artificial. Usually, its resistance is proportional to the light intensity.

- Advantages: usually very inexpensive and very useful
- Disadvantages: cannot discriminate the source or type of light.

Sound sensor

A sound sensor is essentially a microphone that returns a voltage proportional to the ambient noise level. More complex boards can use the data from a microphone for speech recognition.

- Advantages: inexpensive, reliable
- Disadvantages: more meaningful information requires complex programming

Thermal Sensors

Thermal sensors can be used to measure the temperature where it is on a particular component or the ambient temperature.

- Advantages: they can be very accurate
- Disadvantages: more complex and accurate sensors can be more difficult to use.

Thermal Camera

Infrared or thermal imaging allows you to get a complete 2D thermal image of whatever is in front of the camera. This way it is possible to determine the temperature of an object.

- Advantages: differentiate objects from the background based on their thermal signature
- Disadvantages: expensive

Humidity

Humidity sensors detect the percentage of water in the air and are often paired with temperature sensors.

Pressure Sensor

A pressure sensor (which can also be a barometric sensor) can be used to measure atmospheric pressure and give an idea of the altitude of a UAV.

Gas sensors

Specialized gas sensors can be used to detect the presence and concentration of a variety of different gases. However, only specialized robotic applications tend to need gas sensors.

- Advantages: These are the only sensors which can be used to accurately detect gas
- Disadvantages: inexpensive sensors may give false positives or somewhat inaccurate readings and should therefore not be used for critical applications.

Magnetometers

Magnetic sensors or magnetometers can be used to detect magnets and magnetic fields. This is useful to know the position of magnets.

- Advantages: can detect ferromagnetic metals.
- Disadvantages: some times the sensors can be damaged by strong magnets.

Attitude
(roll, pitch and heading)

Compass

A digital compass is able to use the earth's magnetic field to determine its orientation with respect to the magnetic poles. Tilt compensated compasses account for the fact that the robot may not be perfectly horizontal.

- Advantages: provides absolute navigation
- Disadvantages: greater accuracy increases the price

Gyroscope

Electronic gyroscopes are able to provide the angle of the tilt in one or more axes. Mechanical tilt sensors usually determine if a robot has been tilted past a certain value by using mercury in a glas capsule or a conductive ball.

- Advantages: electronic tilt sensors have a higher accuracy than mechanical ones
- Disadvantages: can be expensive

Accelerometers

Accelerometers measure the linear acceleration. This allows to measure the gravitational acceleration or any other accelerations the robot is subject to. This can be a good option to approximate distance travelled if your robot cannot use the surrounding environment as a reference. Accelerometers can measure accelerations along one, two or three axis. A three-axis accelerometer can be used also to measure the orientation a

- Advantages: they do not require any external reference or marker to function and can provide absolute orientation with respect to gravity, or relative orientation.
- Disadvantages: they only approximate the traveled distance and cannot precisely determine it.

IMU's

An Inertial Measurement Unit combines a multi-axis accelerometer with a multi-axis gyroscope and sometimes a multi-axis magnetometer in order to more accurately measure roll

- Advantages: it is a very reliable way of measuring the robots attitude without using external references (besides the earth's magnetic field)
- Disadvantages: can be very expensive and is complex to use.

Miscellaneous

Current and Voltage Sensors

Current and voltage sensors do exactly as their name describes; they measure the current and/or voltage of a specific electric circuit. This can be very useful for gauging how much longer your robot will operate (measure the voltage from the battery) or if your motors are working too hard (measure the current).

- Advantages: they do exactly what they are intended to do
- Disadvantages: can disturb the voltage or current they are measuring. Sometimes they require the circuit being measured to be modified.

Magnetic Sensors

Magnetic sensors or magnetometers detect magnetic objects and can either require contact with the object, or be relatively close to an object. Such sensors can be used on an autonomous lawn mower to detect wire embedded into a lawn.

- Advantages: usually inexpensive
- Disadvantages: usually need to be relatively close to the object, and sadly cannot detect non-magnetic metals.

Vibration

Vibration sensors detect the vibration of an object by using piezoelectric or other technologies.

RFID

Radio Frequency Identification devices use active (powered) or passive (non-powered) RFID tags usually the size and shape of a credit card, small flat disc or addition to a key chain (other shapes are possible as well). When the RFID tag comes within a specific distance of the RFID reader, a signal with the tag's ID is produced.

- Advantages: RFID tags are usually very low cost and can be individually identified
- Disadvantages: not useful for measuring distance, only if a tag is within range.

Practical Examples

1. "I want my robot to follow a person"

(More info on the robot featured in the video…)

There is no "person following sensor" available (yet), so you would need to see which categories above may apply and which don't need to be considered.

- Q: Are you looking to detect, measure distance to (or contact with) an object?
  - Immediately the answer should be yes and this first category of sensors will likely give the best results.
- Q: Are you looking to measure rotation?
  - Perhaps, but you really don't need to know if the robot is rotated (that's a different aspect entirely) or if the human is rotated with respect to the robot.
- Q: Are you looking to measure environmental conditions?
  - Not really. You might consider tracking a human based on their thermal signature, but differentiating between humans and animals (or even a microwave) would be difficult.
- Q: Are you looking to measure position, orientation, or angle?
  - GPS is the first sensor which immediately stands out.

Having gone through the main categories, we should be considering sensors related to distance, contact and detection, and also considering GPS. Taking a closer look at the types of sensors in this category:

- Contact: irrelevant since the robot will be following the human at a distance.
- Distance:
  - Ultrasonic, infrared and laser: measuring the distance is useful when combined with other sensors.
  - Camera: This may be the best option and we will look into it.
  - Stretch: This would require the human to be physically connected to the robot, which is something we do not want.
- Rotation: irrelevant
- Positioning:
  - GPS: placing a GPS unit on both the robot and the human would allow the robot to easily follow the human within a certain radius.
- Environmental conditions: irrelevant
- Attitude:
  - Accelerometer: not very useful since it does not give the robot an idea of where the human is.
  - IMU: not very useful since it does not give the robot an idea of where the human is.
- Miscellaneous:
  - RFID: An RFID reader can locate a tag placed around it, and although some sort of RFID option may be possible, it would require quite a bit of research.

Therefore out of the options available, the most appropriate sensors to allow a robot to follow a human may be ultrasonic or infrared distance sensor(s), a camera and GPS. A camera may be used to pick up a specific pattern placed on the shirt of the individual to follow while GPS units mounted on the robot and on the human would help the robot find the human if she cannot be seen visually. Distance sensors would ensure the robot does not get too close to the human. Therefore when choosing sensors to help your robot follow a human, the sensors listed above would be a good starting point.

2. "I want my robot to stay within the boundaries of our lawn"

There is no "neighbour's grass" sensor available (that we are aware of), so you will need to devise another sensor-based solution.

- Q: Are you looking to detect, measure distance to (or contact with) an object?
    - Yes, we are looking to detect a boundary
- Q: Are you looking to measure rotation?
    - Not really
- Q: Are you looking to measure environmental conditions?
    - Not really, but we'll keep an open mind since the robot is outdoors.
- Q: Are you looking to measure position, orientation, or angle?
    - Not really

Applicable categories therefore include measuring distance, feel contact, detect an object, and perhaps environmental conditions. Out of the list of sensors in this category, we can see that the following may be useful:

- Contact: Detecting collisions in order to avoid obstacles.
- Distance:
    - Ultrasonic, infrared and laser: These will help the robot to avoid hitting objects, and when several placed facing downwards, will help the robot avoid falling into openings such as pools.
- Rotation:
    - Encoders: Encoders: these will help position the robot in two dimensional space based on a starting position.
    - Positioning:
        - GPS: Ideal, the robot could be instructed to remain within certain coordinates.
- Environmental conditions:
    - Humidity sensor: This is not an "intuitive" solution and was creatively used on the Lawnbott Spyder lawn mower to differentiate between grass and "non-humid" surfaces such as concrete and pavement.
    - Magnetic sensor: Magnetic sensors are used both indoors and outdoors to mark boundaries. The perimeter is marked with a strip of conductive wire and the robot is equipped with a few magnetic sensors.
- Attitude:

    - IMU: this may make the data obtained from the encoders more accurate, especially if there are slopes or uneven terrain.
- Miscellaneous: irrelevant

Therefore if you want your robot to stay within the boundaries of your lawn, the sensors listed above would be a very good start.

 For further information on learning how to make a robot, please visit the RobotShop Learning Center. Visit the RobotShop Community Forum in order to seek assistance in building robots, showcase your projects or simply hang-out with other fellow roboticists.

Tags: Grand Tutorial Series Tutorial

Lesson 8

Lessons Menu:

At this stage, you should have all the main components for your robot including actuators, motor controllers, a microcontroller, sensors, and communication systems.

Image credit: zatalian

You are now approaching the integration stage where you will put all these parts together in what will likely be a custom robotic frame. For this, you will need to get your workshop/laboratory/bat cave ready with the appropriate tools.

Robotics Workshop

We have set up three possible robotic-oriented labs scenarios. Choosing which parts to add to your lab depends on how many robots you plan to make, and how involved in robotics you would like to get. We have outlined three broad categories for labs, but don't assume the three labs are exclusive; in the real world, you will undoubtedly find robot builders who have tools from more than one section, and can give you a list of other tools which they have found useful.

The **Essential** setup is intended for first time robot builders who foresee building a few inexpensive robots for fun or have a single project in mind. It is the least expensive setup at less than $100, but don't be fooled by the price tag. In the right hands, a workshop such as this can be used to create professional robots too.

The **Intermediate** setup is intended for builders who are not quite "professional" but are willing to invest a bit more in tools and equipment in order to ease fabrication, assembly, testing and troubleshooting.

The **Ultimate** setup is intended for users who plan to make many advanced robots and prototypes, using a variety of parts and materials. This type of builder wants the finished prototype to look as professional as possible and may even want to produce some small production runs of the finished design. This is the type of setup would likely find at a small robotics company. We cannot cover all the tools required at this level but can give some general suggestions.

As always, it is very important to have the right tool for the right task and only you know your needs best. Below, you will find the various tools and materials suggestions for your workshop classified by level and type.

Mechanical Tools

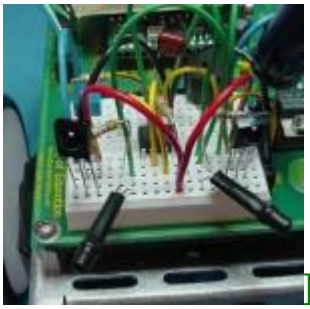
- ## Essential
  - **Small screwdriver set**These small screwdrivers are necessary when working with electronics. Don't force them too much though – their size makes them more fragile.
  - **Regular screwdriver set**All workshops need a multi-tool or tool set which includes flat / Phillips and other screwdriver heads.
  - **Needle nose pliers**A set of needle nose pliers is incredibly useful when working with small components and parts and is a very inexpensive addition to your toolbox. These are different from regular pliers because they come to a point which can get into small areas.
  - **Wire strippers/cutters**If you are planning to cut any wires, a wire stripper will save you considerable time and effort. A wire stripper, when used properly, will only remove a cable insulation and will not produce any kinks or damage the conductors. The other alternative to a wire stripper is a pair of scissors, though the end result can be messy.
  - Scissors, ruler, pen, marker pencil, exacto knife (or other handheld cutting tool)These are essentials in any office.

- ## Intermediate
  - **Rotary Tool**(Dremel for example)Rotary tools have proven to be incredibly versatile and can replace most of the conventional power tools provided the work that needs to be done is at a small-scale. They can cut, drill, sand, engrave, polish, etc.
  - **Drill**A drill is very useful especially when creating larger holes or using stronger / thicker materials. If you are prepared to make the investment, a drill-press allows you to reliably create perfectly perpendicular holes.

- **Saw**A saw of some type is beneficial at this stage to cut thicker materials or make long straight cuts. You can use a hand saw (although you may need to finish the edges), a bandsaw, table saw, etc.
- **Vise**As your work become more complex, you will need to hold materials and parts firmly in place while you work on them. A vise is essential for this and allows to go further in terms of precision and quality.

- **Ultimate**
  - **Tabletop CNC mill**A tabletop CNC machine allows you to precisely machine plastics, metals and other materials and creates three dimensional, intricate shapes.
  - **Tabletop lathe**A (manual) tabletop lathe allows you to create your own hubs, shafts, spacers, adapters and wheels out of various materials. A CNC lathe tends to be overkill since most builders only need to change the diameter rather than create complex shapes.
  - **Vacuum Forming Machine**Vacuum forming machines are used to create complex plastic shells that are moulded to your exact specifications.
  - **Metal Benders**When making robotic frames or enclosures out of sheet metal or metal extrusions, using a metal bender essential in order to obtain precise and repeatable bends.
  - **Other Specialized tools**At this stage, you will be very aware of your machnining needs and will probably require more specialized tools such as metal nibblers, welding machines, 3D printers, etc.

Electrical Tools



- **Essential**
  - **Breadboard**This has nothing to do with slicing bread. These boards are used to easily create prototype circuits without having to solder. This is good in the event that you have not fully developed your soldering skills or want to quickly put together prototypes and test ideas without having to solder a new circuit each time.

- **Jumper wires**These wires fit perfectly from hole to hole on a solderless breadboard and not only look pretty but also prevent clutter.
- **Breadboard power supply** When experimenting with electronics it is very important to have a reliable and easy to use power source. A breadboard power supply is the least expensive power supply offering these features.
- **Soldering tool kit**An inexpensive soldering iron kit has all the basic components needed to help you learn how to solder and make simple circuits.
- **Multimeter**A multimeter is used to measure voltage, resistance, current, check continuity of connections and more. If you know you will be building several robots and working with electronics, it is wise to invest in a higher quality multimeter.
- **Wall adapter**Standard voltages used in robotics include: 3.3V, 5V, 6V, 9V, 12V, 18V and 24V. 6V is a good place to start since it is often the minimum voltage for DC gear motors and microcontrollers and is also the maximum voltage for servo motors. A wall adapter can also be a good replacement for batteries since they can be very expensive in the long run. A wall adapter can allow you to use your project without interruption whereas even rechargeable batteries need to be recharged.

- ## Intermediate
  - The Intermediate electronics lab builds upon the essential lab by adding the following:
  - **Adjustable temperature soldering station**A basic soldering iron can only take you so far. A variable temperature soldering iron with interchangeable tips will allow you to be more precise and decrease the risk of burning or melting components.
  - **Brass sponge for solder**In combination with the more traditional wet sponge to wipe away excess solder, a brass sponge can help clean the soldering iron tip without cooling it down, allowing you to spring back into action quicker and solder like a ninja.
  - **Variable power supply**(instead of wall adapter)Having a powerful and reliable power source is very important when developing complex circuits and robots. A variable power supply allows you to test various voltages and currents without the hassle of needing several types of batteries and power adaptors.

- **Ultimate**
  - **Oscilloscope**An oscilloscope is very useful when dealing with analogue circuits or periodic signals.
  - **Logic Analyser**A logic analyzer is like a “digital eye” when working with digital signals. It allows you to see and store the data produced by a microcontroller and makes it simpler to debug digital circuits.

Miscellaneous

- **Essential**
  - **22 gauge hook-up wire**The most common wire diameter (gauge) used in robotics is 22 (0.0254 ” or 0.64 mm). Although there are advantages to multi-strand wires, single strand (solid core) allows you to easily plug them into pin headers and breadboards.
  - **Third hand**When soldering, having a helping hand that is impervious to heat is extremely useful. A third had is an incredibly helpful tool since it holds the PCB and components in place while you solder.
  - **Hot glue gun**A hot glue gun is incredibly useful no matter what your level of expertise and will only set you back a few dollars. The glue which comes out of a hot glue gun sets rapidly and provides a good bond. Unlike normal glue, this glue is three-dimensional, which means you can use it as a spacer; glue; filler; bridge etc.
  - **Tape**The most popular types of tape used in robotics are duct and electrical. Electrical tape is best suited for electrical components (since it does not conduct) while duct tape is best for structural elements.

- 

Intermediate

  - **Thicker wire**As you build larger robots, DC motors will require higher current and therefore larger diameter wires. The lower the gauge, the thicker the wire and the more current it can handle.
  - **Vernier calliper**In addition to a regular ruler, a vernier allows you to more precisely measure parts as well as diameters (both inside and outside).

Software

- **Essential**

  - **CAD** **Google SketchUp**is a free program which can be used to create your robot in 3D, to the proper scale, complete with texture. This can help you ensure that parts are not overlapping, check dimensions for holes and change the design before it is built.**Autodesk 123D** is another free 3D CAD (Computer Aided Design) software aimed at hobbyists. While it shares many of the same features as Google Sketchup, it has some interesting features such as solid-based part design, assemblies, parametrized transforms and other functionalities that are usually seen in higher end CAD programs.
  - **Programming software**Your first programming software should correspond to whichever microcontroller you selected. If you chose an Arduino microcontroller, you should choose the Arduino software; if you chose a Basic Stamp from Parallax, you should choose PBasic and so forth. In order to use a variety of microcontrollers, you may want to learn a more fundamental programming language such as BASIC or C.
  - **Schematics and PCBs**There are many free programs available on the market, and CadSoft's EAGLE is one of the more popular. It includes an extensive library of parts and helps you convert your schematic to a PCB.

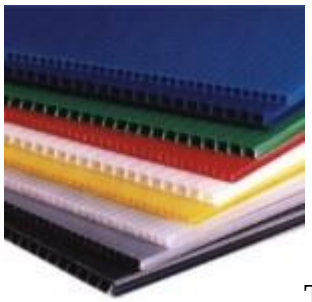
- **Ultimate**
  - **CAD**SolidWorks is the CAD program of choice for many when doing mechanical design but it is certainly not the only one available. Whet working at this level

(i.e. using programs worth several thousands of dollars) you should have a good idea of your needs in order to choose the right tool (Unigraphics, Catia, ProE etc.).

- **CAM**If you are using a CNC machine, you will need a proper 3D CAD program such as ProE, AutoCAD, SolidWorks or other similar program. In order to convert your CAD model to useable code to send to the CNC machine, you need a CAM program. Often you can purchase a CAM program specifically for the CAD software you selected, or find a third-party supplier.

Raw Materials

- ## Essential



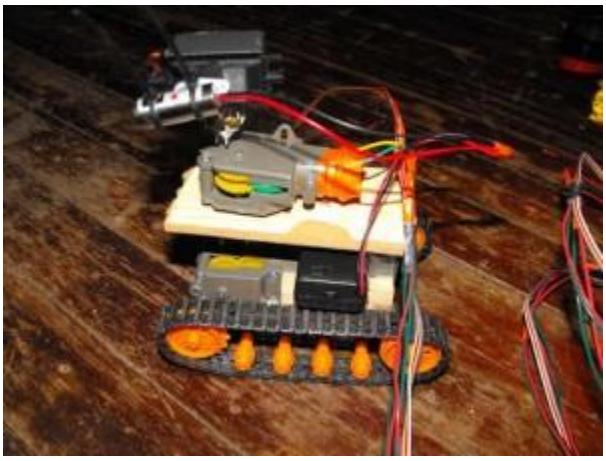
- **Thin sheet metal**This material can be cut easily with scissors and can be bent and shaped as needed to form the frame or other components of your robot without necessarily having to do machining.
- **Cardboard**The right cardboard (thick but can still be cut using hand tools) can easily be used to make a frame or prototype. Even basic glue can be used to hold cardboard together.
- **Thin plastic**Polypropylene, PVC about 1/16" thick can be scored or sawed to create a more rigid and longer lasting frame for your robot.
- **Thin wood**Wood is a great material to work with if you have the means. It can be screwed, glued, sanded, finished and more.

- ## Intermediate

- **Polymorph**Polymorph allows you to create plastic parts without the hassle of having to create custom moulds.
- **Sheet metal**If you have thicker metal-cutting sheers, sheet metal makes an excellent building material for a robot frame because of its durability, flexibility and resistance to rust.
- **Plastic sheets**Plastic sheets are fairly rigid and resist deformation. If you are cautious and slow when cutting or drilling most plastics, the results can look professional

Practical Examples

Essential Workshop: Ard-e



Ard-e, the Arduino based robot , is an example of what you could do achieve with a simple workshop including only essential tools.

Intermediate Workshop: POLYRO

POLYRO is a very advanced robot that can be built with an intermediate workshop. It has most of the features professional robotic platforms used in research laboratories have. Although it has many complex parts, mostly all of them can be put made using simple hand tools.

For the standard practical example included at the bottom of every lesson, only an intermediate level lab would be needed to put the robot together. We will go into more detail in the following lesson.

Ultimate Workshop: BaR2D2

The BaR2D2 is a good example of what can be achieved with such an advanced robotic workshop. It has many intricate custom-machined parts and requires good tooling abilities

For further information on learning how to make a robot, please visit the RobotShop Learning Center. Visit the RobotShop Community Forum in order to seek assistance in building robots, showcase your projects or simply hang-out with other fellow roboticists.

Tags: Grand Tutorial Series Tutorial

Lessons Menu:

Now that you have chosen all the basic building blocks used to make a robot, the next step is to design and build a structure or frame which keeps them all together and gives your robot a distinct look and shape.

Making the Frame

There is no “ideal” way to create a frame since there is almost always a trade-off to be made; You may want a lightweight frame but it may need to use expensive materials or end up too fragile. You may want a robust or large chassis but realize it will be expensive, heavy or hard to produce. Your “ideal” frame may be complex and take too much time to design and create when a simple frame may have been just as good. There is also rarely ever an “ideal” shape, but some designs can certainly look more elegant in their simplicity, while others can attract attention because of their complexity.

Materials

There are many materials you can use to create a frame. As you use more and more materials to build not only robots but other devices, you will get a better feeling as to which is most appropriate for a given project. The list of suggested building materials below include only the more common ones, and once you have tried a few, feel free to experiment with ones not on the list, or merge some together.

Use existing commercial products

 You have likely seen school projects which were based on existing mass produced products such as bottles, cardboard boxes, Tupperware, etc. This is essentially "re-purposing" a product and has the potential to either save you a lot of time and money, or create added hassle and headache. The amazing RoboBrrdto the left is a very good example of how to repurpose materials and make a very capable robot out of them.

Basic construction material

 Some of the most basic construction materials can be used to make excellent frames. One of cheapest and most readily available materials is cardboard, which you can often find for free and can be easily cut, bent, glued and layered. Example: You can create a reinforced cardboard box which looks a lot nicer and is more proportional in size to your robot. You can then spread epoxy or glue to make it more durable and then paint it.

Flat structural material

 One of the most common ways to make a frame is to use a standard material such as a sheet of wood, plastic or metal, and add holes for connecting all the actuators and electronics. A durable piece of wood tends to be fairly thick and heavy, whereas a thin sheet of metal may be too flexible. Example: A flat ⅛" piece of dense wood can be easily cut with a saw, drilled (without fear of shattering), painted, sanded, and more. You can connect devices to both sides (for example connect the motors and caster wheels to the bottom, and the electronics and battery to the top) and the wood will still remain intact and solid.

Laser cut / bent plastic or metal

If you are at the stage where you are prepared to have a frame outsourced, the best options are still to have the part precision cut using a laser or water jet. Having a company produce a custom part is ideal only if you are confident in all your dimensions, since mistakes can be quite costly. Companies which offer computer controlled cutting services many also offer a variety of other services including bending and painting.

3D printing

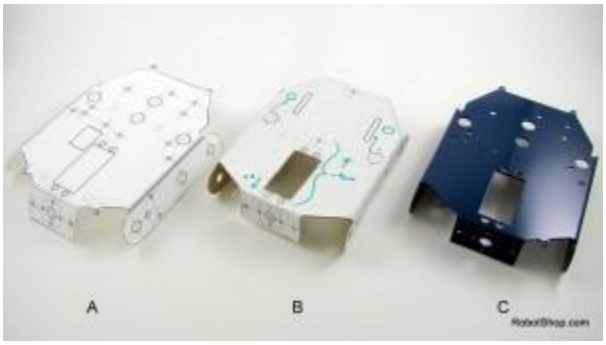

3D printing a frame is rarely ever the most structurally sound solution (because it is built up in layers), but this process can produce very intricate and complex shapes which would not be possible (or very difficult) by other means. A single 3D printed part can contain all the necessary mounting points for all electrical and mechanical components while saving considerable weight. As 3D printing becomes more popular, the price of producing parts will also go down. A very prominent advantage of 3D printing is not only that your design is easy to reproduce, it is also, easy to share. For instance, you can click on the turtle shell example on the left and get all the design instruction and CAD files.

Polymoph

Polymorph is really in a class on its own; at room temperature, Polymorph is a hard plastic, but when heated (in hot water for instance), it becomes malleable and can be shaped into intricate parts, which then cools and solidifies into durable plastic parts. Normally, plastic parts require high temperatures and molds, making them off-limits to most hobbyists. Example: You can combine different shapes (cylinders, flat sheets etc) to form complex plastic structures which look production. You can also experiment with basic molding, the Polymorph robotic arm is a good example of what you can achieve with this material.

Putting the Robot Together

Given the selection of materials and methods, how do you get started? Follow the steps below to create an aesthetic, simple and structurally sound smaller sized robot frame.



1. Settle on a construction material choice.
2. Get all the parts that you robot will require (electrical and mechanical) and measure them. If you don't have all your parts on hand, you can refer to the dimensions provided by manufacturer's
3. Brainstorm and sketch a few different designs for the frame. Don't go into too much detail.
4. Once you settle on a design, make sure the structure is sound and that the components would be well supported.
5. Draw each part of your robot in paper or cardboard at 1:1 scale (real size). You can also draw them using CAD software and print them out.
6. Test your design in CAD and in real life with your paper prototype by test fitting each part and connections.
7. Measure everything again! and once you are absolutely sure your design is correct, start cutting the frame into the actual material. Remember, measure twice and cut once!
8. Test fit each component before assembling the frame in case modifications are require.
9. Go crazy and assemble your frame using hot glue, screws, nails, Duck tape or whatever other binding technique you choose for your robot.
10. Fit all the components onto the frame and voila: you have just created a robot from scratch!

Assembling the Robot Components

Step 10 from the list above deserves to be elaborated upon. In previous lessons, you had chosen the electrical components and actuators. Now, your need to get them all working together. For the following section we will use generic cable colors and terminal names that only cover the common case. As always, the datasheet and manuals are you bests friends when understanding how robotic equipment works.

Connecting Motors to Motor Controllers

A DC (gear) motor, or DC linear actuator will likely have two wires: red and black. Connect the red wire to the M+ terminal on the DC motor controller, and the black to M-. Reversing the wires

will only cause the motor to spin in the opposite direction. A servo motor, there are are three wires: one black (GND), red (4.8 to 6V) and, yellow (position signal). A servo motor controller has pins matching these wires so the servo can be plugged directly to it.

Connecting Batteries to a Motor Controller or a Microcontroller

Most motor controllers have two screw terminals for the battery leads labelled B+ and B-. If your battery came with a connector and your controller uses screw terminals, you may be able to find a mating connector with pigtails (wires) which you can connect to the screw terminal. If not, you may need to find another way to connect the battery to the motor controller while still being able to unplug the battery and connect it to a charger. It is possible that not all the electromechanical products you chose for your robot can operate at the same voltage and thus may require several batteries or voltage regulation circuits. See bellow the usual voltage levels involved in common hobby robotics components:

- DC gear motors – 3V to 24V
- Standard Servo motors – 4.8V to 6V
- Specialty Servo motors – 7.4V to 12V
- Stepper motors – 6V to 12V
- Microcontrollers usually include voltage regulators – 3V to 12V
- Sensors – 3.3V, 5V and 12V
- DC motor controllers – 3V to 48V
- Standard batteries are 3.7V, 4.8V, 6V, 7.4V, 9V, 11.1V and 12V

If you are making a robot with DC gear motors, a microcontroller and maybe a servo or two, it is easy to see how one battery may not be able to power everything directly. We recommend nevertheless, choosing a battery which can directly power as many devices as possible. The battery with the greatest capacity should be associated with the drive motors. For example, if the motors you chose are rated a nominal 12V, your main battery should also be 12V, then you can use a regulator o power a 5V microcontroller. Without going into details, NiMH and LiPo are the top two choices for small to medium-sized robots. Choose NiMh for a cheaper price and LiPo for a lighter weight. Warning: Batteries are powerful devices and can easily burn your circuits if they are connected incorrectly. Always triple check that the polarity is right and that you device can handle the energy provided by the battery. If you are not sure, don't "guess". Electricity is much faster than you, by the time you realize something is wrong, the magic blue smoke already escaped your device.

Connecting Motor controllers to Microcontroller

A microcontroller can communicate with motor controllers in a variety of ways:

- Serial: The controller has two pins labelled Rx (receive) and Tx (transmit). Connect the Rx pin of the motor controller to the microcontroller's Tx pin and vice versa.

- I2C: The motor controller will have four pins: SDA, SCL, V, GND. Your microcontroller will have the same four pins but not necessarily labelled, simply connect them one to one.
- PWM: The motor controller will have both a PWM input and a digital input for each motor. Connect the PWM input pin of the motor controller to a PWM output pin on the microcontroller, and connect each digital input pin of the motor controller to a digital output pin on the microcontroller.
- R/C: To connect a microcontroller to an R/C motor controller, you need to connect the signal pin to a digital pin on the microcontroller.

Regardless of the communication method, the motor controller's logic and the microcontroller need to share the same ground reference (this is achieved by connecting the GND pins together) and the same logic high level (this can be achieved by using the same V+ pin to power both devices). A logic level shifter is required if the devices don't share the same logic levels (3.3V and 5V for instance)

Connecting Sensors to a Microcontroller

Sensors can be interfaced with microcontrollers in a similar way than motor controllers. Sensors can use the following types of communication:

- Digital: The sensor has a digital signal pin that connects directly to a digital microcontroller pin. A simple switch can be regarded as a digital sensor.
- Analogue: Analogue sensors produce an analogue voltage signal that needs to be read by an analogue pin. If your microcontroller does not have analog pins, you will need a separate analog to digital circuit (ADC). Also, some sensors some with the required power supply circuit and usually have three pins: V+, GND and Signal. If a sensor is a simple variable resistor for instance, it will require you to create a voltage divider in order to read the resulting variable voltage.
- Serial or I2C: the same communication principles explained for motor controllers apply here.

Communication device to microcontroller

Most communication devices (e.g. XBee, Bluetooth) use serial communication, so the same RX,TX, GND and V+ connections are required. It is important to note that although several serial connections can be shared on the same RX and TX pins, proper bus arbitration is required in order to prevent cross-talk, errors and madness in general. If you have very few serial devices, it is often simple to use a single serial port for each one of them.

Wheels to motors

Ideally, you would have chosen wheels or sprockets which are designed to fit the shaft of the motor you chose. If not, hopefully there is a hub which fits between the two. If you find that the wheel and motor you have chosen are not compatible with one another and cannot find a suitable

hub, you may need to find another hub which connects to the wheel but has a smaller bore, you would then drill out the hub's bore to the same diameter as the shaft.

Electrical components to frame

You can mount electronics to a frame using a variety of methods. Be sure that whatever means you use do not conduct electricity. Common methods include: hex spacers, screws, nuts, double-sided tape, Velcro, glue, cable ties, etc.

Practical Example

1. Settle on a construction material choice.
2. We are getting the following parts in order to measure and test fit them:
    - 2x Gear motors with 90 degree shafts
    - 1x Arduino Uno
    - 1x Johnny Robot Standard GM Track Kit
    - 2x Arduino Motor controller Shield
    - 2x Mini Breadboard
3. We will try to stay close to a 6 sided box, but may have had to make modifications in order to accommodate for all parts
4. Some modifications need to be done to the design in order to accommodate for all parts such as:
    - Add more mounting holes for the battery pack
    - Add more mounting points for servos or other accessories
    - Refined the hole placement.
5. The cardboard frame will be made by printing the design onto white cardboard (or gluing a printed paper sheet onto cardboard), cutting it, bending it and using (hot) glue in order to reinforce the bends, edges and surfaces.
6. We completely assembled the robot using the cardboard frame in order to make sure everything fits properly.
7. We measure everything again and once we were absolutely sure about the design, we had it professionally manufactured.
8. Test fit each component in case modifications are require.
9. The frame is made in one piece so no assembly is required
10. Assembled the robot incorporating lots of accessories.

 For further information on learning how to make a robot, please visit the RobotShop Learning Center. Visit the RobotShop Community Forum in order to seek assistance in building robots, showcase your projects or simply hang-out with other fellow roboticists.

Tags: Grand Tutorial Series How To Make a Robot

Lesson 10

Programming is usually the final step involved in building a robot. If you followed the lessons, so far you have chosen the actuators, electronics, sensors and more, and have assembled the robot so it hopefully looks something like what you had initially set out to build. Without programming though, the robot is a very nice looking and expensive paperweight.

It would take much more than one lesson to teach you how to program a robot, so instead, this lesson will help you with how to get started and where (and what) to learn. The practical example will use "Processing", a popular hobbyist programming language intended to be used with the Arduino microcontroller chosen in previous lessons. We will also assume that you will be programming a microcontroller rather than software for a full-fledged computer.

What Language to Choose?

 There are many programming languages which can be used to program microcontrollers, the most common of which are:

- **Assembly**; its just one step away from machine code and as such it is very tedious to use. Assembly should only be used when you need absolute instruction-level control of your code.
- **Basic;** one of the first widely used programming languages, it is still used by some microcontrollers (Basic Micro, BasicX, Parallax) for educational robots.
- **C/C++;** one of the most popular languages, C provides high-level functionality while keeping a good low-level control.
- **Java**; it is more modern than C and provides lots of safety features to the detriment of low-level control. Some manufacturers like Parallax make microcontrollers specifically for use with Java.

- **.NET**/**C#**; Microsoft's proprietary language used to develop applications in Visual Studio. Examples include Netduino, FEZ Rhino and others).
- **Processing** (Arduino); a variant of C++ that includes some simplifications in order to make the programming for easier.
- **Python**, one of the most popular scripting languages. It is very simple to learn and can be used to put programs together very fast and efficiently.

In lesson 4, you chose a microcontroller based on the features you needed (number of I/O, user community, special features, etc). Often times, a microcontroller is intended to be programmed in a specific language. For example:

- Arduino microcontrollers use Arduino software and are re-programmed in Processing.
- Basic Stamp microcontrollers use PBasic
- Basic Atom microcontrollers use Basic Micro
- Javelin Stamp from Parallax is programmed in Java

If you have chosen a hobbyist microcontroller from a known or popular manufacturer, there is likely a large book available so you can learn to program in their chosen programming language. If you instead chose a microcontroller from a smaller, lesser known manufacturer (e.g. since it had many features which you thought would be useful for your project), it's important to see what language the controller is intended to be programmed in (C in many cases) and what development tools are there available (usually from the chip manufacturer).

Getting Started

The first program you will likely write is "Hello World" (referred to as such for historic reasons). This is one of the simplest programs that can be made in a computer and is intended to print a line of text (e.g. "Hello World") on the computer monitor or LCD screen. In the case of a microcontroller, another very basic program you can do that has an effect on the outside world (rather than just on-board computations) is toggling an IO pin. Connecting an LED to and I/O pin then setting the I/O pin to ON and OFF will make the LED blink. Although the simple act of turning on an LED may seem basic, the function can allow for some complex programs (you can use it to light up multi-segment LEDs, to display text and numbers, operate relays, servos and more).

Step 1: Ensure you have all components needed to program the microcontroller

Not all microcontrollers come with everything you need to program them, and most microcontrollers need to be connected to a computer via USB plug. If your microcontroller does

not have a USB or DB9 connector, then you will need a separate USB to serial adapter, and wire it correctly. Fortunately many hobbyist microcontrollers are programmable either via an RS-232 port or by USB, and include the USB connector on-board which is used not only for two-way communication, but also to power the microcontroller board.

Step 2: Connect the microcontroller to the computer and verify which COM port it is connected to. Not all microcontrollers will be picked up by the computer and you should read the "getting started" guide in the manual to know exactly what to do to have your computer recognize it and be able to communicate with it. You often need to download "drivers" (specific to each operating system) to allow your computer to understand how to communicate with the microcontroller and/or the USB to serial converter chip.

Step 3: Check product's user guide for sample code and communication method / protocol

Don't reinvent the wheel if you don't have to. Most manufacturers provide some code (or pseudo code) explaining how to get their product working. The sample code may not be in the programming language of your choice, but don't despair; do a search on the Internet to see if other people have created the necessary code.

- Check product manuals / user guides
- Check the manufacturer's forum
- Check the internet for the product + code
- Read the manual to understand how to write the code

Useful Tips

1. Create manageable chunks of functional code: By creating segments of code specific to each product, you gradually build up a library. Develop a file system on your computer to easily look up the necessary code.
2. Document everything within the code using comments: Documenting everything is necessary in almost all jobs, especially robotics. As you become more and more advanced, you may add comments to general sections of code, though as you start, you should add a comment to (almost) every line.
3. Save different versions of the code – do not always overwrite the same file: if you find one day that your 200+ lines of code do not compile, you won't be stuck going through it line by line; instead you can revert to a previously saved (and functional) version and add / modify it as needed. Code does not take up much space o a hard drive, so you should not feel pressured to only save a few copies.
4. Raise the robot off the table or floor when debugging (so its wheels/legs/tracks don't accidentally launch it off the edge), and have the power switch close by in case the robot tries to destroy itself. An example of this is if you try to send a servo motor to a 400us signal when it only accepts a 500 (corresponding to 0 degrees) to 2500us (corresponding to 180 degrees) signal. The servo would try to move to a location which it cannot physically go to (-9 degrees) and ultimately burn out.

5. If code does something that does not seem to be working correctly after a few seconds, turn off the power – it's highly unlikely the problem will "fix itself" and in the meantime, you may be destroying part of the mechanics.
6. Subroutines may be a bit difficult to understand at first, but they greatly simplify your code. If a segment of code is repeated many times within the code, it is a good candidate to be replaced with a subroutine.

Practical Example

We have chosen an Arduino microcontroller to be the "brain" of our robot. To get started, we can take a look at the Arduino 5 Minute Tutorials. These tutorials will help you use and understand the basic functionality of the Arduino programming language. Once you have finished these tutorials, take a look at the example below.

For the robot we have made, we will create code to have it move around (left, right, forward, reverse), move the two servos (pan/tilt) and communicate with the distance sensor. We chose Arduino because of the large user community, abundance of sample code and ease of integration with other products.

Distance sensor

Fortunately in the Arduino code, there is an example for getting values from an analog sensor. For this, we go to File -> Examples -> Analog -> AnalogInOutSerial (so we can see the values)

Pan/Tilt

Again, we are fortunate to have sample code to operate servos from an Arduino. File -> Examples -> Servo -> Sweep

Note that text after two slashes // are comments and not part of the compiled code

```
#include <Servo.h>          // This loads the servo script, allowing you to use specific functions below

Servo myservo;                  // create servo object to control a servo
int pos = 0;                    // variable to store the servo position

void setup()                    // required in all Arduino code
{
myservo.attach(9);          // attaches the servo on pin 9 to the servo object
}

void loop()                     // required in all Arduino code
{
for(pos = 0; pos < 180; pos += 1)  // variable ‘pos’ goes from 0 degrees to 180 degrees in steps of 1 degree
{
myservo.write(pos);             // tell servo to go to position in variable ‘pos’
delay(15);                      // waits 15ms for the servo to reach the position
}
for(pos = 180; pos>=1; pos-=1)     // variable ‘pos’ goes from 180 degrees to 0 degrees
{
myservo.write(pos);             // tell servo to go to position in variable ‘pos’
delay(15);                      // waits 15ms at each degree
}
}
```

Motor Controller

Here is where it gets a bit harder, since no sample code is available specifically for the Arduino. The controller is connected to the Tx (serial) pin of the Arduino and waits for a specific "start byte" before taking any action. The manual does indicate the communication protocol required; a string with specific structure:

- 0x80 (start byte)
- 0x00 (specific to this motor controller; if it receives anything else it will not take action)
- motor # and direction (motor one or two and direction explained in the manual)
- motor speed (hexadecimal from 0 to 127)

In order to do this, we create a character with each of these as bytes within the character:

```
unsigned char buff[6];

buff[0]=0x80; //start byte specific to Pololu motor controller
buff[1]=0; //Device type byte specific to this Pololu controller
buff[2]=1; //Motor number and direction byte; motor one =00,01
buff[3]=127; //Motor speed "0 to 128″ (ex 100 is 64 in hex)

Serial.write(buff);
```

Therefore when this is sent via the serial pin, it will be sent in the correct order.

Putting all the code together makes the robot move forward and sweep the servo while reading distance values.

You can see the full robot and the user manual.

VJ Hirsch Computer/Math Instructor

Van Nuys High School

6536 Cedros Avenue

Van Nuys, CA 91411

vxh16221@lausd.net

818.778.6800