Autonomous Mobile Robots

Lecture 03: Motors

Lecture is based on material from Robotic Explorations: A Hands-on Introduction to Engineering, Fred Martin, Prentice Hall, 2001.

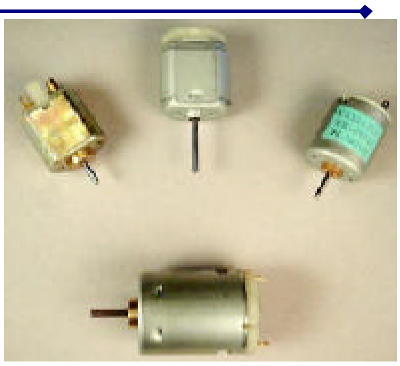
Outline

- DC Motors
- Gearing
- Electronic Control
- The Servo Motor
- LEGO Design

Homework #3

- **Motors:** Read Chapter 4 of Robotic Explorations (textbook)
- **Construction Techniques:** Read Appendix B of Robotic Explorations (textbook)
- **LEGO Design:** Read "The Art of LEGO Design," by Fred Martin (linked on course web page)

DC Motors



Direct Current (DC) Motors:

- Small, cheap, reasonably efficient, easy to use, ideal for small robotic applications
- Converts electrical energy into mechanical energy
- How do they work?
  - By running electrical current through loops of wires mounted on rotating shaft (*armature*)
  - When current is flowing, loops of wire generate a magnetic field, which reacts against the magnetic fields of permanent magnets positioned around the wire loops
  - These magnetic fields push against one another and the armature turns
- Efficiency
  - Various limitations, including mechanical friction, cause some electrical energy to be wasted as heat
  - Toy motors: efficiencies of 50%
  - Industrial-grade motors: 90%

DC Motors

Properties:

- **Operating Voltage**
  - Recommended voltage for powering the motor
  - Most motors will run fine at lower voltages, though they will be less powerful
  - Can operate at higher voltages at expense of operating life
- **Operating Current**
  - When provided with a constant voltage, a motor draws current proportional to how much work it is doing
  - When there is no resistance to its motion, the motor draws the least amount of current; when there is so much resistance as to cause the motor to stall, it draws the maximal amount of current
  - *Stall current:* the maximum amount of operating current that a motor can draw at its specified voltage

DC Motors

Properties:

- **Torque**
  - Rotational force that a motor can deliver at a certain distance from the shaft
    - The more current through a motor, the more torque at the motor's shaft
  - Direct consequence of the electromagnetic reaction between the loops of wire in the motor's armature and the permanent magnets surrounding them
  - Strength of magnetic field generated in loops of wire is directly proportional to amount of current flowing through them; torque produced on motor's shaft is a result of interaction between these two magnetic fields
  - Often a motor will be rated by its *stall torque*, the amount of rotational force produced when the motor is stalled at its recommended operating voltage, drawing the maximal stall current at this voltage
  - Typical torque units: *ounce-inches*; e.g., 5 oz.-in. torque means motor can pull weight of 5 oz up through a pulley 1 inch away from the shaft

DC Motors

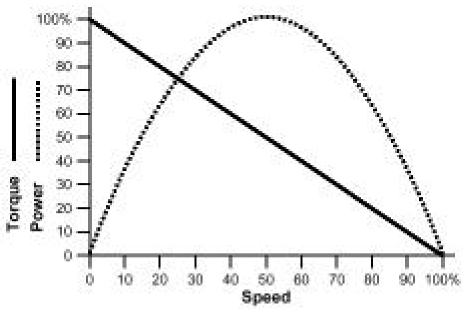
Properties:

- **Power**
  - Product of the output shaft's *rotational velocity* and torque
  - Output Power Zero
    - Case 1: Torque is zero
      - Motor is spinning freely with no load on the shaft
      - Rotational velocity is at its highest, but the torque is zero—it's not driving any mechanism (Actually, the motor is doing some work to overcome internal friction, but that is of no value as output power)
    - Case 2: Rotational Velocity is zero
      - Motor is stalled, it is producing its maximal torque
      - Rotational velocity is zero
  - In between two extremes, output power has a characteristic parabolic relationship

DC Motors

Motor Speed vs. Torque, Power:

- **Solid line** shows the relationship between motor speed and torque
  - At the right of the graph, the speed is greatest (100%) and the torque is zero; this represents the case where the motor shaft is spinning freely but doing no actual work
  - At the left of the graph, the speed is zero but the torque is at its maximum; this represents the case where the shaft is stalled because of too much load
- **Dashed line** shows the power output, which is the product of speed and torque
  - It is the highest in the middle of the motor's performance range, when both speed and torque are produced



Idealized Graph

DC Motors

Measuring Motor Torque:

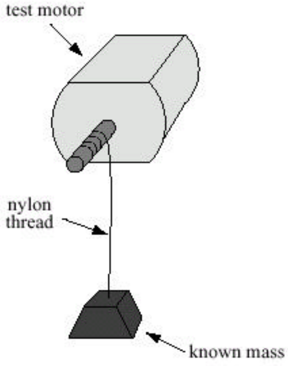
• Motor winds a nylon thread, which carries a known weight, around the motor shaft

• As the thread winds up around the shaft, like a bobbin, the effective radius of the shaft increases

• This process continues until the radius of the bobbin increases to a point where the motor can no longer lift the weight.

• When the motor stops turning, measure the radius of the bobbin

• **Stall torque = bobbin radius * mass**



Experiment:
Bobbin radius = 0.5 in.
Mass = 2 ounce
Torque = 1 ounce-inch

DC Motors

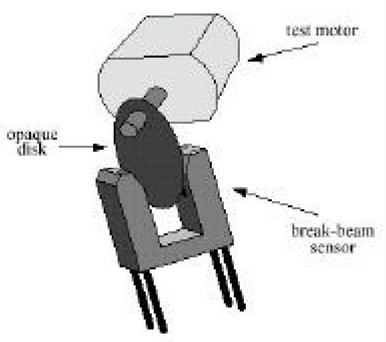
Measuring Motor's Top Speed in RPM:

• Opaque disk (light-weight) is mounted directly on the motor shaft

• *Break-beam opto-sensor* is positioned such that as the disk rotates, it interrupts the sensor's light beam once per revolution

• For counting the transitions on the sensor, use *pulse accumulator input* (PAI = sensor input #9), which counts pulses on a particular digital input pin with hardware ancillary to the 6811 core (allows very fast rate, transparent to the rest of the processor's functioning)

• Most DC motors have unloaded speeds in the range of 3,000 to 9,000 revolutions per minute (RPM), which translates to between 50 and 150 revolutions per second. This is slow enough that a regular 68HC11 analog input could be used, but it is possible that Interactive C would not be able to keep up with this rate.



Experiment:
Use torque or RPM test to determine if motor is symmetric in both directions

DC Motors

Measuring Motor's Top Speed in RPM:

• **rpm.c** uses the poke, bit set,and bit clear operations to manipulate two 68HC11 registers, the PACTL (Port A ConTroL) and PACNT (Port A CouNT) to perform the pulse-accumulation function

• Main loop of the program runs for 6 seconds, incrementing a hi count variable when the PACNT register overflows (it is only one byte long, and thus can only count from 0 to 255)

• Total number of counts is multiplied by ten to yield a measurement of revolutions per minute

```
/* rpm.c */
int PACTL= 0x1026;   /* pulse accumulator control */
int PACNT= 0x1027;   /* pulse accumulator count */
int PAEN= 0x40;      /* bit to enable counting */
int rpm()
{
    long end_time;
    int hi_count= 0;
    int last_count= 0;
    bit_set(PACTL, PAEN); /* enable counting */
    poke(PACNT, 0);      /* reset to 0 */
    end_time= mseconds() + 6000L; /* 6 sec */
    while (mseconds() < end_time) {
        if (peek(PACNT) < last_count) hi_count++;
        last_count= peek(PACNT);
    }
bit_clear(PACTL, PAEN);  /* disable counting */
/* report result in revolutions per minute */
return 10 * (hi_count * 256 + last_count);
}
```

Experiment:
Use rpm.c to measure the speed of various DC motors

Gearing

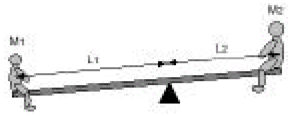
• DC motors are high-speed, low-torque devices

• All mechanisms in robots, including drive trains and actuators, require more torque and less speed

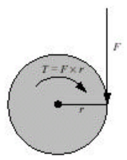
• **Gears** are used to trade-off high speed of the motor for more torque

• Torque, or rotational force, generated at the center of a gear:

$$T = F \times r$$



Downward force is equal to weight times their distance from the fulcrum. Lighter people can displace heavier people simply by increasing their distance from the fulcrum.



The torque t—or, turning force—is the product of a force F applied perpendicularly at a radius r.

Gearing

Meshing Gears

• When two gears of unequal sizes are meshed together, their respective radii determine the translation of torque from the driving gear to the driven one

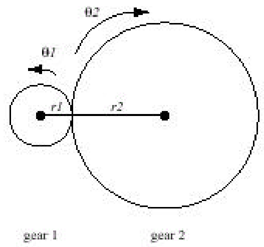
• This mechanical advantage is easiest understood from a “conservation of work” point of view

W = F x d

W = T x q

• Neglecting losses due to friction, no work is lost or gained when one gear turns another

• Example: Gear 1’s radius is one-third that of Gear 2. Their circumferences are also in a 3:1 ratio, so it take three turns of the small gear to produce one turn of the larger gear. Ratio of resulting torques is also 3:1.



Gear 1 with radius r1 turns an angular distance of θ1 while **Gear 2** with radius r2 turns an angular distance of θ2.

Ratio of gear sizes determines ratio of resulting torques

 13

Gearing

Gear Reduction

• Small gear driving a larger one:

– **torque increases**

– **speed decreases**

Exercise: calculate effective gear ratio of *HandyBug*’s drive train

• 3 to 1 Gear Reduction

– Power applied to 8-tooth gear results in 1/3 reduction in speed an 3 times increase in torque at 24-tooth gear

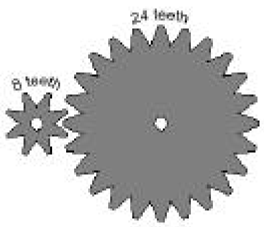
• 9 to 1 Gear Reduction

– By putting two 3:1 gear reductions in series—or “ganging” them—a 9:1 gear reduction is created

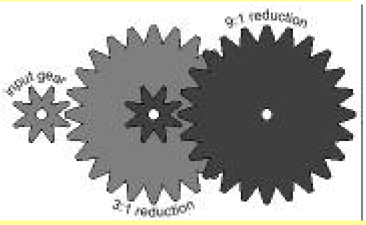
– The effect of each pair of reductions is multiplied to achieve the overall reduction

– Key to achieving useful power from a DC motor

– With this gear reduction, the high speed and low torque is transformed into usable speeds and powerful torques



3 turns of left gear (8 teeth) to cause 1 turn of right gear (24 teeth)



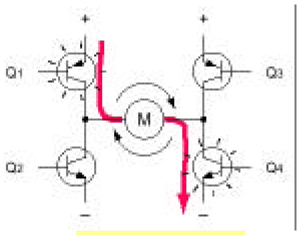
8-tooth gear on left; 24-tooth gear on right

 14

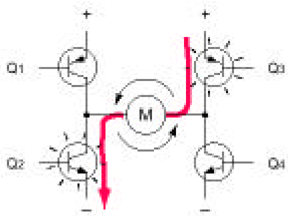
Electronic Control

H-Bridge Motor Driver Circuit

- Four transistors form the vertical legs of the H, while the motor forms the crossbar
- In order to operate the motor, a diagonally opposite pair of transistors must be enabled
- Transistors **Q1 and Q4 enabled**
    - Starting with the positive power terminal, current flows down through Q1, through the motor from left to right, down Q4, and to the negative power terminal
    - Results in motor rotating in a clockwise direction
- Transistors **Q2 and Q3 enabled**
    - Results in current flowing through the motor from right to left



Q1 and Q4 enabled

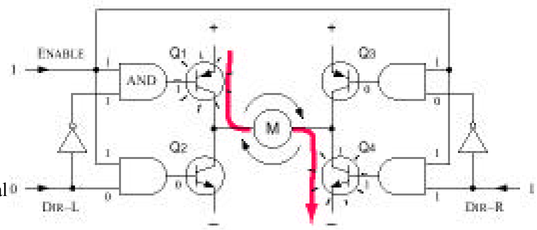


Q2 and Q3 enabled

Electronic Control

Enable and Direction Logic

- Critical that transistors in either vertical leg of "H" are never turned on at same time
    - If Q1 and Q2 were turned on together, current would flow straight down through the two transistors
    - There would be no load in this circuit other than the transistors themselves, so the maximal amount of current possible for the circuit would flow, limited only by the power supply itself or when the transistors self-destructed
- Actual circuit has hardware to facilitate control of transistor switches
    - Add four AND gates and two inverters
    - AND gates accept enable signal that allows one signal to turn whole circuit on/off
    - Inverters ensure that only one transistor in each vertical leg of the H is enabled at any one time



DIR–L =0, DIR–R=1, enable signal =1: Q1 and Q4 turn on, and current flows through the motor from left to right

DIR–L =1, DIR–R=0, enable signal =1: Q2 and Q3 turn on, and current flows through the motor from in the reverse direction

Electronic Control

Active Braking

• What happens if both direction bits are the same state, and the enable bit is turned on?

– *Effectively, both terminals of motor are connected together*

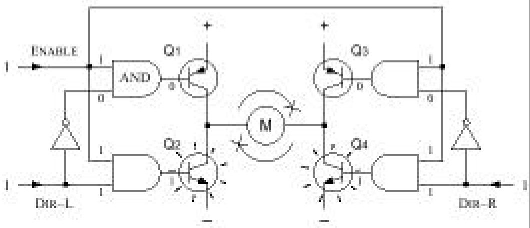
• Motor acts as a generator, creating electricity

–If there is a load connected to the motor, then the motor resists being turned proportional to the amount of the load

– When the motor terminals are grounded through the transistors, it is as if the motor were driving an infinite load

– Transistors in the H-bridge act as a wire connecting the motor terminals— the infinite load

• Final result: circuit acts to actively brake the motor's spin; transistors absorb the energy generated by the motor and cause it to stop. If, on the other hand, none of the transistors is active, then the motor is allowed to spin freely; i.e., to coast



Both direction bits are one and the enable bit is turned on causing transistors **Q2 and Q4** to be activated. This causes both terminals of the motor to be tied to the voltage supply less the voltage drop of the transistor (0.6v).

Contemporary **electric car** designs incorporate circuitry to convert the the drive motor into a generator for recharging the main batteries when braking. This way, the power stored in the car's motion is recovered back into electrical energy. The **active braking** doesn't apply enough force to replace conventional brakes, but it can significantly extend the electrical car's operating range.

Electronic Control

Speed Control

• ***Pulse Width Modulation* (PWM)**

– The H-bridge circuit allows control of a motor's speed simply by turning the drive transistor pair on and off rapidly

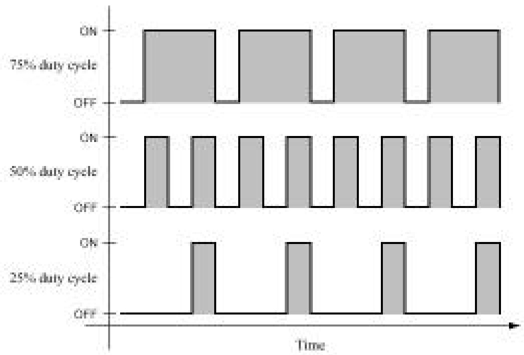
– *Duty cycle*—proportion between "on time" and "off time"—determines fractional amount of full power delivered to motor

– Commonly used in practice: simpler to build circuits that switch transistors on and off than to supply varying voltages at the currents necessary to drive motors

– Tends to be fairly linear (25% duty cycle yields pretty close to one-quarter of full power)

• **Reducing the voltage applied to the motor**

– Giving a motor 1/4 of its normal operating voltage typically would result in much less than 1/4 of nominal power, since the power increases approximately as the square of the voltage



PWM works by rapidly turning the motor drive power on and off. Waveforms shown would be connected directly to the enable input. Three sample **duty cycles** are shown: a 75%, a 50%, and a 25% rate. The frequency used in PWM control is generally not critical. Over a fairly wide range, from between 50 Hz and 1000 Hz, the motor acts to average the power that is applied to it.

Electronic Control

HB Implementation

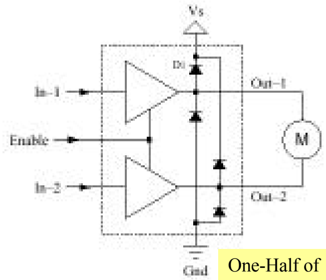
• HB uses two copies of H-bridge driver, either SGS-Thomson L293D or TI SN754410 - chips accept digital logic signals as input and drive motors directly on their outputs

• Each triangular driver replaces one "leg," or two transistors, in the H-bridge circuits. Each driver may be either driven high (enabled and input is high), driven low (enabled and input is low), or turned off (disabled and input doesn't matter).

• To make the motor spin, the enable input must be high, and one driver in-put must be high and the other low. If the enable is high, and both driver inputs are high or both are low, then the circuit actively brakes the motor. If the enable is low, then the motor is allowed to coast.

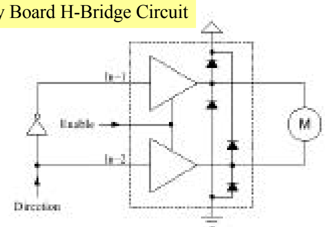
• Rather than individually control IN–1 and IN–2, the Handy Board adds an inverter so that a single bit may be used to determine motor direction. When the direction input is high, then IN–2 is high and IN–1 is low. When the direction is low, IN–2 is low and IN–1 is high.

• The full Handy Board circuit uses a 8–bit latch, the 74HC374 chip, which provides the eight bits necessary to control four motors.



One-Half of L293D/SN754410 Motor Driver Chip

Handy Board H-Bridge Circuit



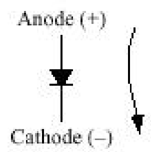
Electronic Control

Spike-Canceling Diodes

• Also part of the motor driver chips are **four diodes** connecting from each driver output to either Vs , the motor voltage supply, or ground. These diodes perform the important function of trapping and shunting away inductive voltage spikes that naturally occur as part of any motor's operation.

• Diodes allow current to flow in one direction only. If there is a higher voltage on the anode than on the cathode, then current flows through the diode

• The diodes in the motor driver chip may appear to be connected backward, but they are drawn correctly. When a motor is running, the coil of wire in its armature acts as an inductor, and when the electricity in this coil changes, voltage spikes are generated that might be of *higher voltage than the* Vs *power supply* or *lower voltage than ground.*



Diode: current flows from higher voltages on the anode to lower voltages on the cathode, in the direction of the diode's arrowhead.

Example: suppose a voltage greater than Vs is generated by the motor on the OUT–1 line. Then the diode labeled D1 conducts, shunting this voltage to the Vs power supply. If the diodes were not present, these inductive voltage spikes would enter the voltage supply of the rest of the project circuitry, possibly doing damage to more sensitive components.

Servo Motor

Specifications

Futaba S148 Servo Motor with Mounting Horns ($17.00)

- Specialized motor for turning to a specific position
- Components:
  - DC motor
  - Gear reduction unit
  - Shaft position sensor
  - Electronic circuit that controls the motor's operation
- "Servo" - capability to self-regulate its behavior, i.e., to measure its own position and compensate for external loads when responding to a control signal
- Widely used in hobby radio control applications:
  - RC cars: position the front wheel rack-and-pinion steering
  - RC airplanes: control the orientation of the wing flaps and rudders
- Positioning applications:
  - Shaft travel is restricted to 180 degrees
  - Input waveform specifies desired angular position of output shaft
  - Electronics measure current position
  - If different from desired position, servo is turned on to drive the shaft to the desired position

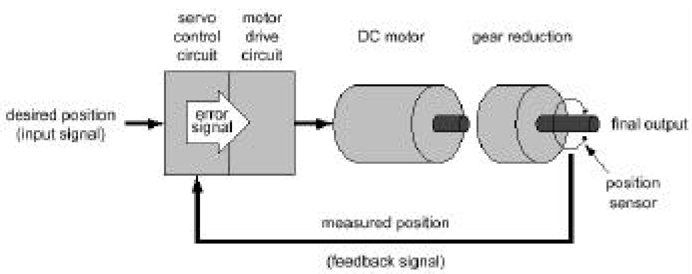
 21

Servo Motor

Servo Control

- Most hobby servo motors use a standard three wire interface:
  - Power
  - Ground
  - Control Line
- Power supply is typically 5 to 6 v

The **input** to the **servo motor** is desired position of the output shaft. This signal is compared with a **feedback signal** indicating the actual position of the shaft (as measured by position sensor). An **"error signal"** is generated that directs the motor drive circuit to power the motor. The servo's gear reduction drives the final output.



 22

Servo Motor

Servo Control Signal

• Control line uses a **PWM scheme** for encoding the position signal

• Servo PWM method is different from the speed control PWM

– Speed control PWM: **overall duty cycle** (i.e, percentage of on-time) determines the speed of the motor

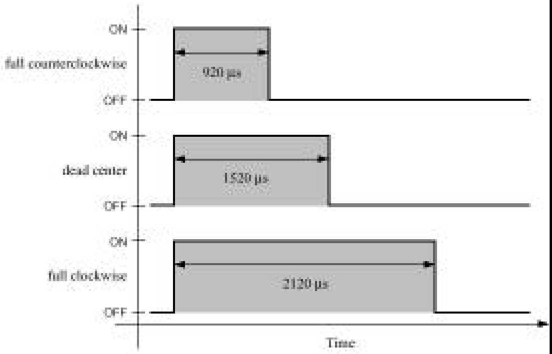
– Servo PWM: **length of the pulse** is interpreted to signify control value

• Waveforms' length:

– **920 ms** - full counterclockwise

– **1520 ms** - center position

– **2120 ms** - full clockwise



Three sample waveforms for controlling a servo motor

Servo Motor

Servo Control Signal

• To complete the servo control, all that one must do is periodically repeat the individual control pulses

• Servo turns off when pulses stop

• For **Futaba servo motors**, the recommended interval between control pulses is 14 to 20 ms

• Servo Timing Signal

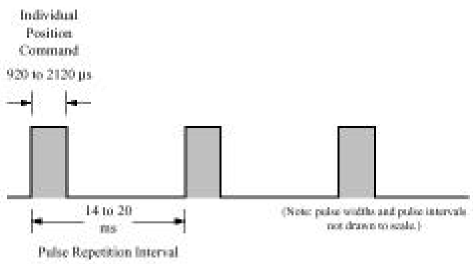
– Pulse width must be accurate in µs; otherwise servo exhibits **jitter**

– Interval between pulses may vary 14 - 20 ms; successive pulses need not be exactly same distance apart

• Limits mechanical first, then electrical

– Electronics will try to drive output shaft to a point beyond mechanical limits

• **Do not plug servo motor in backwards!**



To get the servo motor to continually attempt to reach the desired position, the timing pulse must be repeated at a regular interval

Experiment: find range of motion of different servo motors

Servo Motor

Generating Control Waveform (HB's servo driver software)

- **Driver Routines**
  - **servo_a5.icb** and **servo_a7.icb** each generate waveform to control position of one servo motor
  - Handy Board can thus operate two servo motors simultaneously
- **Driver Pins**
  - **servo_a5.icb** generates signal on **Port A bit 5** (pin 29, timer output #3), also connector header of Expansion Bus, "TO3"
  - **servo_a7.icb** generates signal on **Port A bit 7** (pin 27 one of the timer #1 outputs), also connector header of Digital Input Bank, input 9
- **Software Interface**
  - For each servo, a global variable is set, generating the pulse train which acts as a desired position input to the servo
  - Value of the control global should be set to the twice the length of the desired control pulse
    - e.g., for 1520 µs centering pulse, control global = 3040
  - Values outside workable portion of the range will cause servo to overtax itself as it tries to reach a position that is not mechanically possible

Servo Motor

Generating Control Waveform (HB's servo driver software)

- Two drivers must be loaded into IC, either at the command line or via a **.lis** file.
- **Servo_a7.icb** [Do not load servo.icb (from standard IC library) and servo_a7.icb at the same time]
  - **int servo_a7_pulse** Integer global variable determining value of servo control signal. Units are 0.5 µs counts; e.g., a value of 3040 yields a pulse length of 1.52 ms, which is in the middle of a typical servo's range.
  - The default value of this global is 2560, which is re-established on board reset
  - **int servo_a7_init(int enable)** Function to enable and disable the servo output. Call with argument equal to one to enable, and zero to disable.
- **Servo_a5.icb**
  - **int servo_a5_pulse** Integer global variable determining value of servo control signal. Units are 0.5 µs counts; e.g., a value of 3040 yields a pulse length of 1.52 ms, which is in the middle of a typical servo's range.
  - The default value of this global is 2560, which is re-established on board reset
  - **int servo_a5_init(int enable)** Function to enable and disable the servo output. Call with argument equal to one to enable, and zero to disable.

Servo Motor

Continuous Rotation - "Winch" Servo

- Servo motor's output shaft rotates back and forth with a sweep of travel of about 180 degrees
- ***Winch servo*** rotates continuously
  - – Control signal specifies the speed and direction of rotation, rather than the desired angular position
  - – Useful for wide variety of applications, including robot's main drive motors
- Conversion:
  - – Feedback potentiometer is replaced by a pair of fixed resistors, which mimic the center position of the potentiometer; when the control signal deviates from center, the servo's control electronics drive the motor one way or the other in a vain attempt to get the servo to move away from center
  - – Result is that the servo spins continuously with user-controllable speed and direction
- This methods allows both **speed and direction control**: the farther the control signal is away from the center position, the faster the motor turns

Servo Motor

Exercises for your Servo Motor

1. Calculate the **duty cycle** of the servo control pulse, assuming dead center positioning and pulse intervals of 18 ms.

2. For each variety of servo motor that you have on hand, experimentally determine the **control pulse values** corresponding to the limits of rotary travel.

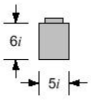
3. Write **wrapper functions** for setting the servo control globals and protecting the servo motors from out of bounds values.

4. Write a function to cause the servo motor to **"sweep"** its position back and forth.

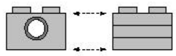
5. With a servo that has been modified for **continuous rotation**, determine how far away from the center position the control signal needs to be to get the motor to run at its maximum speed. What does this tell you about the control function mapping the error signal into motor power?

LEGO Design

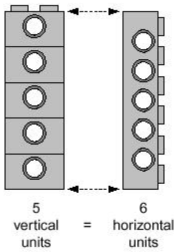
Structure



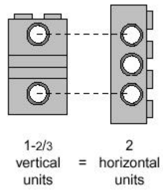
Unit LEGO brick
i is a conversion factor between "LEGO lengths" and standard units
6/5 height full-size brick

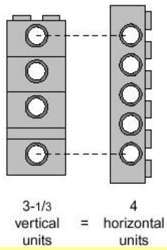


Three of the **thin LEGO plates** are equal in height to the unit brick
2/5 height thin plate



Stack of Five LEGO Bricks = Six-Long LEGO Beam

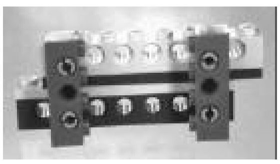




Two-Unit and Four-Unit Vertical LEGO Spacings

LEGO Design

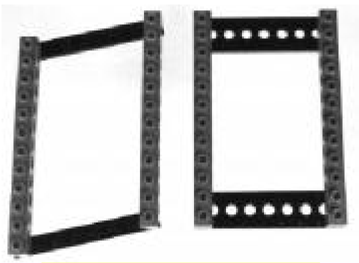
Structure



Sturdy LEGO construction

Full Height Bricks	One-Third Height Plates	Horizontal Units
1	2	2
3	1	4
5	0	6
6	2	8

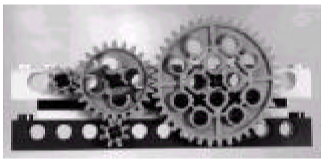
Black peg is slightly larger; fits snugly
Gray peg rotates freely



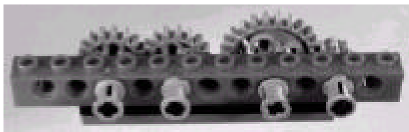
Square Corners: use 2x plates rather than 1x ones

LEGO Design

Gearing

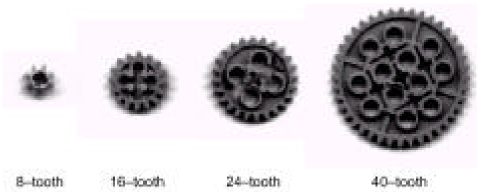


The 8–tooth, 24–tooth, and 40– tooth round gears all mesh properly along a horizontal beam because they have **“half unit” radii.** The 8– and 24– tooth gears are meshed horizontally at two units, and vertically.



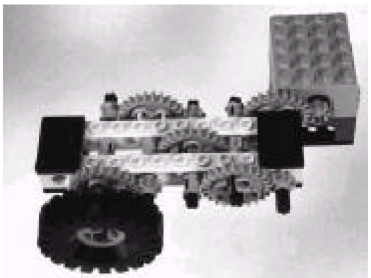
The 16–tooth gear has a radius of **1 LEGO unit,** so two of them mesh properly together at a spacing of two units. Since an 8– and 24– tooth gear also mesh at two-unit spacing, these respective pairs of gears **can be swapped** for one another in an existing geartrain.

Number of Teeth	Radius in Horizontal Units
8	1/2
16	1
24	1–1/2
40	2–1/2

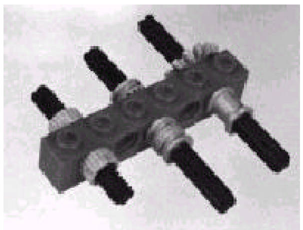


LEGO Design

Gearing



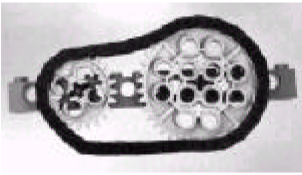
A five-stage reduction using 8– and 24–tooth gears creates a **243-to-1 reduction** in this sample LEGO geartrain. Note the need for three parallel planes of motion to prevent the gears from interfering with one another. Four 2x3 LEGO plates are used to hold the beams square and keep the axles from binding.

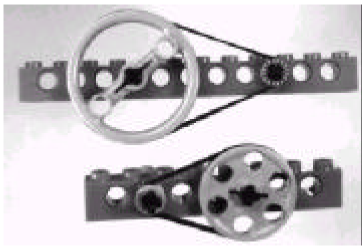


- Standard 1-LEGO-long **stop bush** (upper axle, front) is not the only part that can act as a **bushing** (axle holder)
- Small **pulley wheel** (middle axle) acts as a half-sized spacer—it also grabs tighter than the full bush
- **Bevel gear** (upper axle, back) makes a great bushing
- **Nut-and-bolt parts** (lower axle) can be used to make a tight connection

LEGO Design

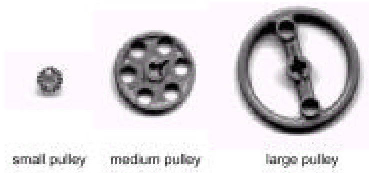
Chain Links and Pulleys





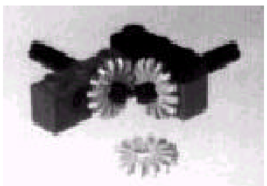
Chain links can be an effective way to deliver large amounts of torque to a final drive, while providing a gear reduction if needed. Chain link works best at the slower stages of gearing, and with a somewhat slack link-age. Use the larger gears—the 8–tooth one won't work very well.

There are three sizes of **pulley wheels**:
- Tiny one, which doubles as a **stop bush**
- Medium-sized one, which doubles as a **tire hub**
- Large-sized one, which is sometimes used as a **steering wheel** in official LEGO plans



LEGO Design

Crown and Bevel Gears

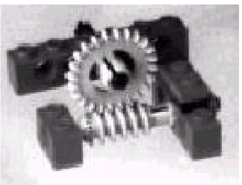


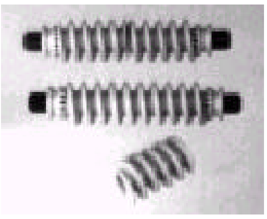
The 8–tooth gear, in conjunction with the 24-tooth **crown gear**, is used to **change the axis of rotation** in a gear train. In this instance, the configuration provides for a **vertical shaft output.** Horizontal output also possible.

The **bevel gears** are used to **change the angle of rotation** of shafts in a gear train with a 1:1 ratio. In this case, they are used to effect a change in the **horizontal plane.**

LEGO Design

Worm Gear





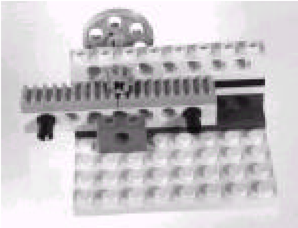
The **worm gear** is valuable because it acts as a **gear with one tooth**: each revolution of the worm gear advances the round gear it's driving by just one tooth. So the worm gear meshed with a 24-tooth gear yields a **24:1 reduction**. The worm gear, however, loses a lot of power to friction, so it may not be suitable for high performance, main drive applications.

- **Bottom** is the basic worm gear, two horizontal LEGO units in length
- **Top** is an unsuccessful attempt to put two worm gears on the same shaft
- **Middle** is the successful attempt

When placing multiple worm gears on a shaft, the trick is to try all four possible orientations to find the one that works.

LEGO Design

Gear Rack



The gear driving the **gear rack** is often referred to as the **"pinion,"** as in "rack-and-pinion steering," which **uses the transverse motion of the gear rack to orient wheels**. The 8–tooth gear is a good candidate to drive the rack be-cause of the gear reduction it achieves—one revolution of the gear moves the rack by eight teeth.

LEGO Design

Geartrain Design Tips

- Work **backward from the final drive**, rather than forward from the motor
  - Usually there is a fair bit of flexibility about where the motor is ultimately mounted, but much less in the placement of drive wheels or leg joints
  - Start by mounting the axle shaft that will carry the final drive, put a wheel and gear on it, and start working backward, adding gearing until there is enough, and finally mount the motor in a convenient spot
- Do not forget about the **role of the tire** in determining the relationship between the rotational speed of the final drive axle and the linear speed that is achieved
  - Small tires act as gear reductions with respect to large tires, and this may have an effect on how much gear reduction is necessary
- If **geartrain performing badly**
  - Make sure the stop bushes are not squeezing too hard—there should be some room for the axles to shift back and forth in their mounts
  - Check that all beams holding the axles are squarely locked together
- To **test a geartrain**, try driving it **backward**
  - If your geartrain can be readily back-driven, it is performing well

LEGO Design

LEGO Clichés (from Fred Martin)

On occasion it is necessary to **lock a beam to an axle**. This figure shows how to use a medium **pulley wheel**, which rigidly locks to an axle, to hold the beam in place.

The special **"gear mounter"** piece is an axle on one side and a loose connector peg on the other. It can be used to mount gears used as idlers in a gear train — used simply to transmit motion or to reverse the direction of rotation.

This configuration of parts can be used as a **compact axle joiner.** LEGO now produces a part designed for this purpose, but in lieu of that part, this is a useful trick.

LEGO Design

LEGO Clichés (from Fred Martin)

In order to **build outward from a vertical wall** of axle holes, a smaller beam may be mounted with its top studs in the holes of the beam wall.

The recommended way to build outward from a beam wall is to use the **connector-peg-with-stud piece,** which is a loose-style connector peg on one end and a top stud on the other.

The **full-size stop bush** can be used in one orientation to **hold an axle through a plate hole** so that the axle can freely rotate.

LEGO Design

LEGO Clichés (from Fred Martin)

By using the stop bush to hold an axle in place between two plates, a **vertical axle mount** can easily be created. Depending on the orientation of the stop bush, it can be made to either lock the axle in place or allow it to rotate freely.

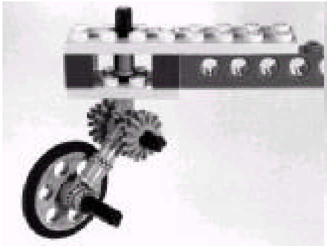
In the other orientation, the stop bush locks between four top studs, perfectly centered over the axle holes in flat plates. This allows the stop bush to **lock a plate to an axle.**

The **"toggle joint"** can be used to lock two axles at a variety of odd angles. The short axle running through the two toggle joints is equipped with stop bushes on either end to hold the joint together.

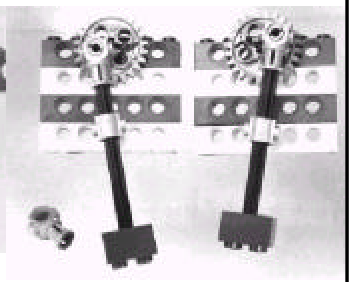
LEGO Design

LEGO Clichés (from Fred Martin)

Here the **toggle joint** is used to connect two axles at right angles. The small pulley wheel is deployed on the axle that runs through the toggle joint to either **lock the axle or allow it to rotate.**



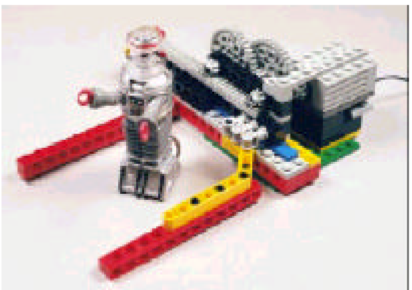
Several clichés are used to construct this **caster wheel**.



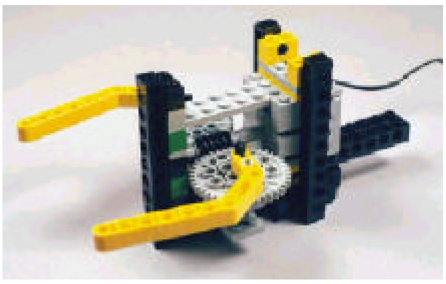
The **"piston rod"** part is used twice in each mechanism to create a **LEGO leg**. By using a chain drive or gear linkage to lock legs in sync, a multi-legged creature can be designed.

LEGO Design

LEGO Clichés (from Fred Martin)



Robot Gripper Using Gear Rack



Robot Gripper Using Worm Gear