Building LEGO® Robots
For
FIRST™ LEGO® League

Version: 1.0
Sept. 23, 2002

By
Dean Hystad

INSciTE
Innovations in Science and Technology Education

www.hightechkids.org

About the Author and This Document

Dean Hystad is a first class LEGO fanatic. Dean is an engineer for MTS Systems located in Eden Prairie, Minnesota. MTS Systems Corporation is one of the worldís leading suppliers of mechanical testing and simulation equipment, making everything from earthquake simulators to amusement park rides. At MTS, Dean writes control software for a range of large robotic systems. As you can see in this book, Dean brings that expertise to his passion for LEGO. Dean has judged at FLL events for several years and is currently working as a contributing author on a soon-to-be published book on LEGO Mindstorms. Dean can be reached at dean.hystad@mts.com.

This book is a significant piece of work, and frankly is awe inspiring to me. It is an excellent mix of theory and practice. Some of the text may be beyond middle school youth, but the accompanying labs and presentation material should help coaches understand the concepts and present key ideas to their teams. The book can also provide a useful text for those of you wanting to incorporate FLL into high school classes. I hope you find it useful.

This book fits well with INSciTE's mission to advance innovative programs that provide an environment where kids, educators, and the technical community come together to cultivate life long learning in science, math and technology.

Fred Rose
Fredrose@hightechkids.org
INSciTE President
September 20, 2002

Copyright and Trademark Notice

INSciTE
PO Box 41221
Plymouth, MN 55441

www.hightechkids.org

Table of Contents

Figures

Tables

1 Structures

1.1 Bricks, Plates, and Beams

Bricks, plates, and beams are not as glamorous as the RCX brick, motors, and sensors. But they are the fundamental components that are used to build the frame that supports the RCX, counteracts the motorís forces, and holds the sensors precisely in place. Master some LEGO building fundamentals first, and your team will have success. Ignore them, and youíll spend more time repairing your robot than you did building it.

1.1.1 Bricks

This is a LEGO building brick. Little has changed since its introduction in 1949. According to LEGO, they have produced 320 billion bricks[1] since that time. Thatís approximately 52 bricks per person living today.

Figure 1-1. Basic LEGO Brick

LEGO bricks are made out of ABS plastic. They are injection molded to very exacting tolerances (0.002mm)[2]. The top of the brick is covered with cylindrical plastic bumps called studs. The bottom of the brick has cylindrical holes or tubes. When you snap two bricks together, the tubes deform slightly around the studs, locking the two firmly together.

1.1.1.1 Dimensions

The common practice is to refer to bricks using their dimensions: width, length, and height (though height is often left off when referring to standard sized bricks). When doing this, the width and length dimensions are given in studs. The piece below is a 2 x 4 brick.

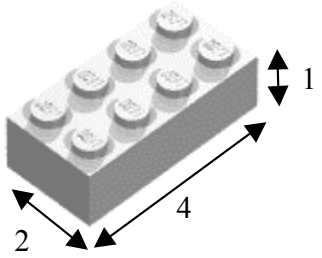


Figure 1-2. Brick Dimensions

LEGO bricks are based on the metric system. The 2 x 4 brick above is 16mm wide, 32 mm long, and 9.6mm high (ignoring the studs on top). That works out to 1 stud = 8mm.

[1] From www.lego.com/eng/info/history
[2] From ì Jin Satoís Lego Mindstorms The Masterís Techniqueî

It also means that bricks are 1.2 studs high. This asymmetry can lead to design and building difficulties as will be discussed later.

Question: What is the smallest sized cube that can be made out of LEGO bricks?

1.1.2 Plates

Plates are essentially short bricks. They are 1/3 the height of standard bricks--3.2mm or 0.4 studs. Plates use the same naming convention as bricks.

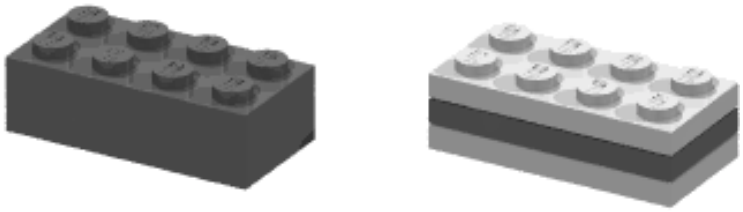


Figure 1-3. Three Plates = One Brick high

Some plates have through holes aligned with the backside tubes. They are referred to as Technic plates, or less obscurely, plates with holes. The holes accept axles and connector pins and make the Technic plates much more useful.

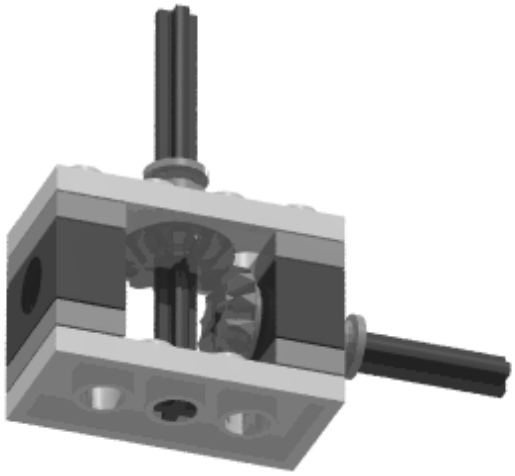


Figure 1-4. Simple Gearbox Using Technic Plates

Construction Note: Use normal plates when you don't need the through holes. Save the Technic plates for where they are needed.

Question: What is the smallest sized cube that can be made out of bricks and plates?

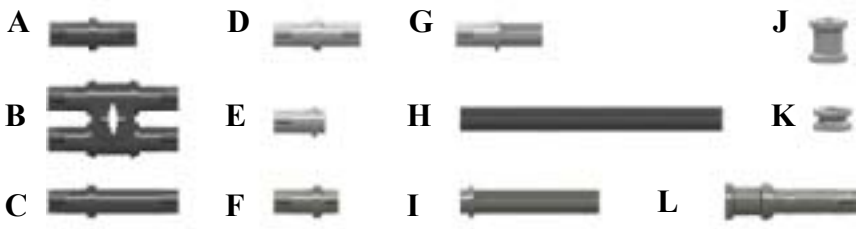
1.1.3 Beams

In 1977, LEGO introduced Technic[3], a series of complex models for older children to build. Central to Technic are the new beams which are 1x bricks with holes in their sides. The holes are spaced at one-stud intervals and centered between the studs on the top of the beam. The beams can be stacked on top of each other just like bricks. In addition, connector pins can be placed in the side holes allowing the beams to be assembled side by side. The number of assembly techniques available using the new parts is staggering.

Figure 1-5. Technic Beams

1.1.4 Axles and Pins

The RIS kit comes supplied with a wide variety of pins and axles for connecting Technic beams together. The most commonly used of these is the black Technic pin with friction, or friction pin. The friction pin has small, raised ridges that make it lock tightly in the holes of a Technic beam providing a very strong connection. A long version of the friction pin can be used to pin three beams together. The double pin works well with the goofy transparent blue connector block that comes in the newer RIS sets and has an axle hole that can sometimes be useful.



A. Pin with Friction
B. 3L Double Pin
C. Long Pin with Friction
D. Technic Pin
E. Half Pin
F. æ Pin
G. Axle Pin
H. 6L Axle
I. 3L Pin with Stud
J. Full Bushing
K. Half Bushing
L. Long Pin with Stop Bushing

Figure 1-6. Pins and Axles

Slightly less common is the gray Technic pin. Similar in appearance to the friction pin, it lacks the ridges and has a slightly loose fit. The Technic pin is a good choice for pivots or hinges. The short post of the æ Technic pin fits nicely in the half deep holes in the side of the RCX, and it is often used to attach beams to the side of the programmable

[3] From www.lego.com/eng/info/history

brick. The short post of the Ω Technic pin is actually the same size as a LEGO stud and can be used to mimic studs coming out of the side of a beam.

Figure 1-7. Studs on the Side of a Beam?

Axles are long rods that have a ì+î shaped cross section. They slide easily through the holes in Technic beams, but they fit tightly in the cross-holes found in wheels, gears, bushings, and other Technic elements. Axles are available in even stud lengths starting at 2 and going up to 12. There are also 3 and 5 stud long axles. It is common to use shorthand when referring to axles, describing the axle using its length followed by ì Lî. Thus a four stud long axle is a 4L axle.

Figure 1-8. A Typical Driven Wheel Assembly

There are a few oddball parts that donít really fit in the axle or pin category. The first is the aptly named axle pin which is half Technic pin and half axle. It is most commonly used to attach gears to the sides of beams. Another is the 3L axle with stud, which is a three stud long Technic axle with a stud on one end. Try sticking the stud in the hole of a Technic beam. The connection is surprisingly strong. The long Technic pin with stop bushing is my favorite part. I use it whenever I need a removable pin connection because the bushing on the end is easy to grasp. The cross-hole in the stop bushing accepts an axle and can be used to make some interesting parts.

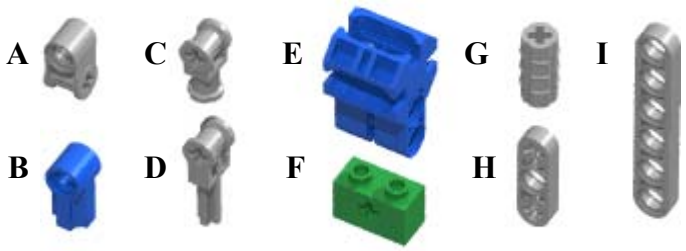
Figure 1-9. A 16L Technic Pin

1.1.5 LEGO Vocabulary

The RIS kit is filled with all kinds of interesting plastic parts that donít look much like anything with which you are used to building. Look hard and youíll not find one screw,

nut, bolt, or nail. Instead, you have the blue holey-thing and the little gray whatchamabob with the axle in it.

To aid in communication and facilitate the exchange of ideas, the LEGO community has come up with names for each thingamajig and doohickey. Most of the names were provided by LEGO--extracted from marketing or packaging literature. But many part monikers originated in the user community. Figure 1-10 lists the names of some of the Technic parts included in the RIS kit.



A. Axle Jointer Perpendicular
B. Angle Connector #1
C. Connector with Axlehole
D. Pole Reverser Handle
E. Connector Block 3 x 2 x 2
F. Beam 1 x 2 with Axlehole
G. Axle Joiner
H. Liftarm 1 x 3
I. Liftarm 1 x 6

Figure 1-10. Common Technic Pieces

Links: Jim Hughes maintains *Technica*, a beautiful website that has a complete Technic parts registry with pictures and an interesting history of LEGO. His URL is w3.one.net/~hughesj/technica/technica.html.

1.2 Building a Frame

A robot needs some sort of frame. The frame gives the robot its shape. It provides mounting points for sensors and reacts the forces generated by motors and gears. It is like our skeleton, which gives us our shape, supports our organs, and reacts the forces generated by our muscles. A good frame is strong, lightweight, and holds together even after much use.

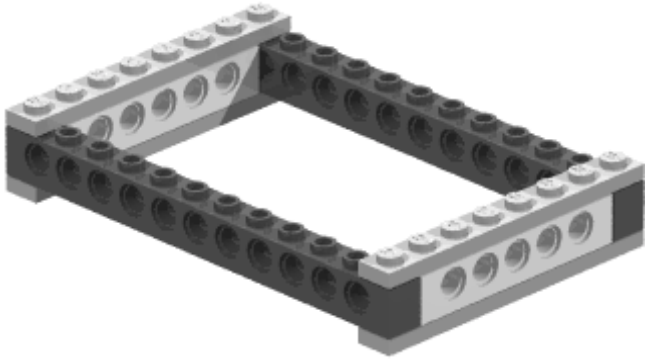


Figure 1-11. Simple Frame

Figure 1-11 shows a simple frame made out of beams and 1x8 plates. Itís strong, lightweight, and the dimensions are appropriate for the base of a robot platform. But it is

not very rigid. A gentle push on opposing corners causes the frame to twist out of shape. Eventually the corner connections work loose, and the frame falls apart.

The problem is that the plates do not lock the corners at right angles. There is a small amount of clearance between the ends of the 1 x 6 beams and the sides of the 1 x 12 beams. This allows the studs to act as hinges. Replacing one or more of the 1 x 8 plates with 2 x 8 plates makes the frame much more rigid.

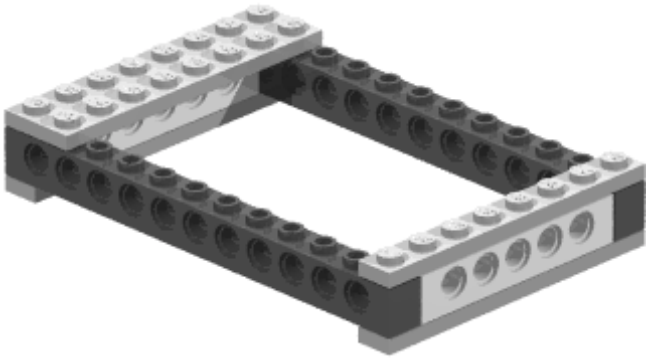


Figure 1-12. Improved Frame

The improved frame is much stiffer. Pushing on the corners causes it to flex hardly at all. The 2 x 8 plate firmly locks the short and long beams together at right angles. This frame is adequate for many applications, but it can be made even stronger.

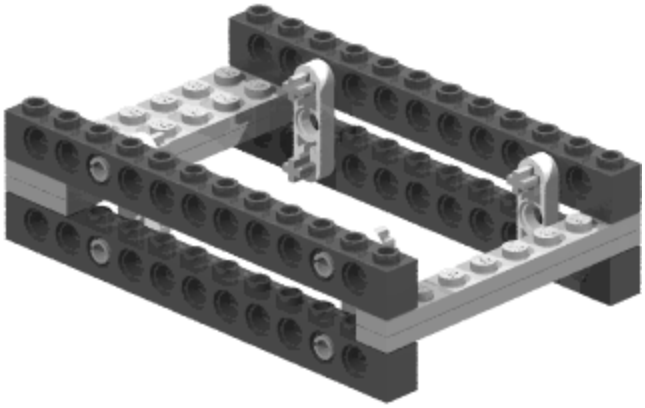


Figure 1-13. Cross Braced Frame

The frame in Figure 1-13 uses cross bracing to hold it together. The 1 x 3 lift arms prevent the frame from being pulled apart. Cross bracing is a useful technique for building very strong LEGO structures. The connections between LEGO bricks are very strong in compression (a force pushing the bricks together), and in shear (a sideways force trying to slide the bricks across each other), but they are relatively weak in tension (a force pulling the bricks apart). When cross bracing, we reinforce a connection which is weak in tension by adding components that will be in shear. In this case, the 1 x 3 lift arms and axle pins are in shear when trying to pull the beams apart.

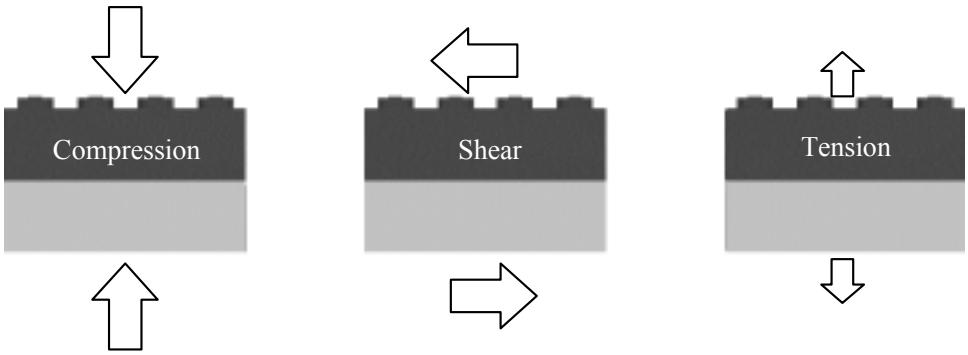


Figure 1-14. Snap On Connections are Weak in Tension

1.2.1 LEGO Geometry

A 1 x 6 Technic beam is 1 stud wide, 6 studs long, and 1.2 studs high. Bricks and beams not being an integer number of studs high causes problems when trying to do cross bracing. This can be seen in the left assembly in Figure 1-15. The second hole in the vertical beam does not line up with the hole in the horizontal beam.

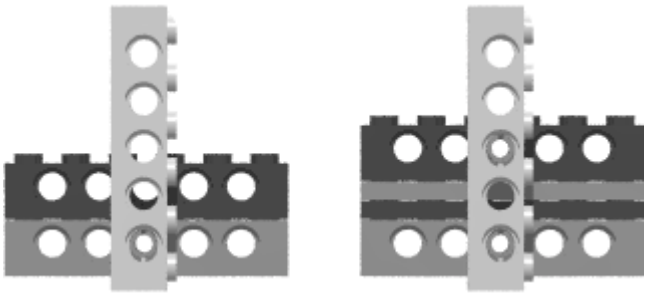


Figure 1-15. Cross Bracing

In the assembly on the right, the third hole in the vertical beam aligns perfectly with the hole in the horizontal beam. This is because the beam and the two plates add up to exactly two studs tall (1.2 + 2 * 0.4 = 2). You will find that cross bracing only works out when the vertical hole spacing is divisible by two.

Question: What is the shortest stack of beams (no plates allowed) that will align with holes in a vertical beam?

Construction Note: The two assemblies below are the same height, but the one on the right allows for locking the beam at an intermediate point and has better spacing for vertical meshing of LEGO gears.

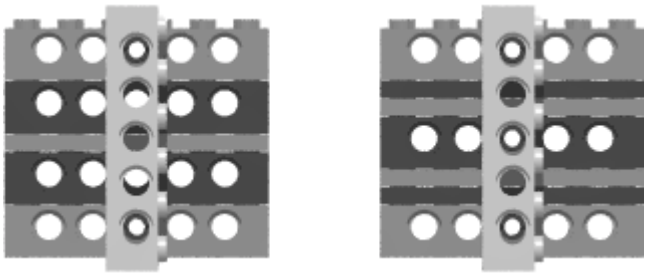


Figure 1-16. Two Cross Bracing Choices

This same technique can be used to build tall frames that are lightweight and strong. Figure 1-17 shows a tall frame that uses friction pins at the corners to attach the vertical and horizontal members. Notice that the 1-2-1 (1 Beam-2 Plates-1 Beam) technique is used to get the proper spacing.

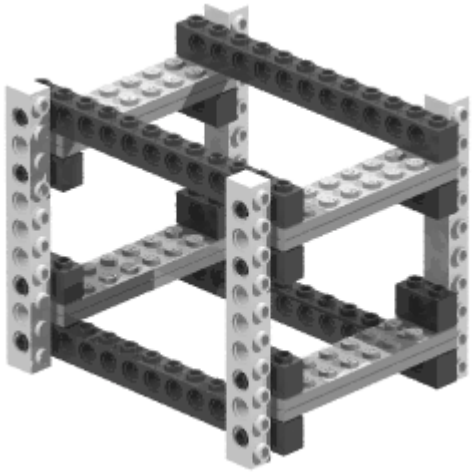


Figure 1-17. A Tall Frame

1.2.1.1 Diagonal Bracing

Itís very easy to get locked into the mindset that horizontal and vertical are the only ways to build. The grid-like nature of the LEGO pieces reinforces this thinking. But diagonal connections are possible as well.

Diagonal bracing is trickier to implement than perpendicular cross bracing. Cross bracing can be used on any assembly where the dimension is evenly divisible by two studs, but it canít be used anywhere else. With diagonal bracing there are more solutions, but their derivations are not as obvious.

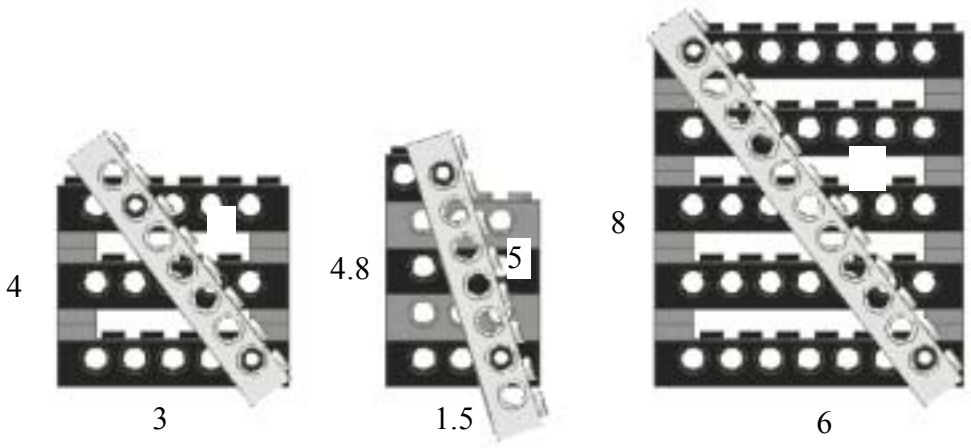


Figure 1-18. Diagonal Cross Bracing

You can find diagonal bracing solutions through experimentation. Lay the diagonal brace on the part, and move it about until you find a place where the holes line up. It can also be done analytically using the Pythagorean Theorem for right triangles. ì The sum of the squares of the legs of a right triangle equals the square of the hypotenuse.î This is often written as ì $C = \sqrt{A^2 + B^2}$.î The two legs are the base (width across the bottom) and the height. The hypotenuse is the diagonal beam. Diagonal bracing is possible if the hypotenuse is close to an integer number (less than 0.05 studs difference).

Table 1-1. Calculating Diagonal Lengths

Base (A)	Height (B)	Hypotenuse	Comments
4	4	5.65	Doesnít fit
3	4	5	Perfect fit
1.5	4.8	5.03	Fits, but a little tight
6	8	10	Perfect fit

1.3 SNOT

SNOT is all the rage in the LEGO building community today. SNOT, standing for Studs Not On Top, is a way of building with LEGO that does not always utilize the traditional click-fit assembly techniques. This can be very tricky and requires great imagination, but the resulting models are very beautiful and quite realistic.

Figure 1-19. One of Jennifer Clark's Incredible Creations. Yes itís LEGO.

Luckily, Technic is more flexible to build with than other LEGO. But it is still difficult sometimes to figure out a way to attach the motor where you want it or to position the sensor in just the right place. Trying to discover new ways to put LEGO together is a challenging activity that can be a lot of fun. Here are some ideas to get the creative juices flowing.

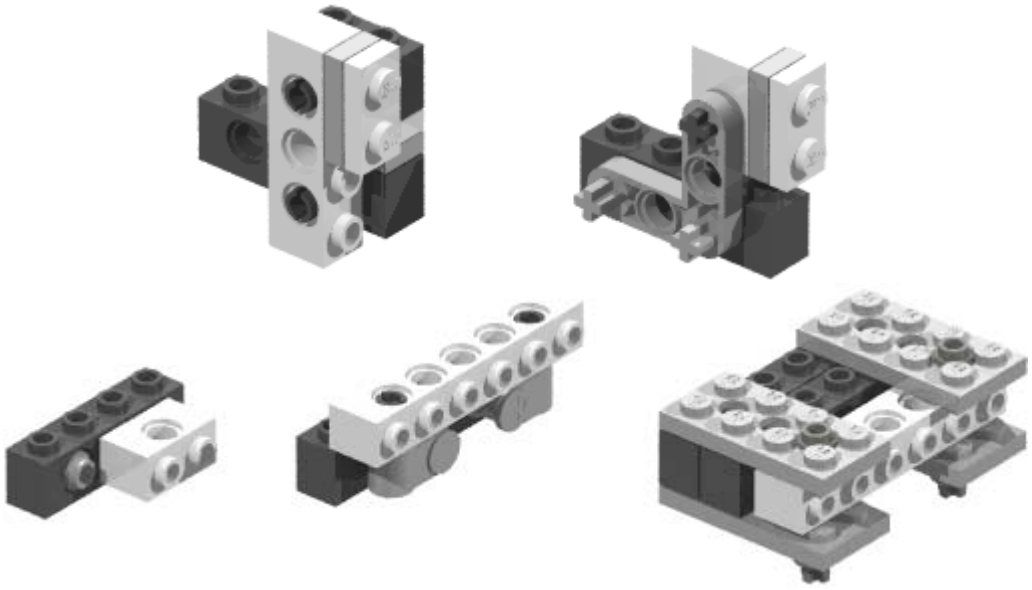


Figure 1-20. Turning 90 degrees. Studs Out

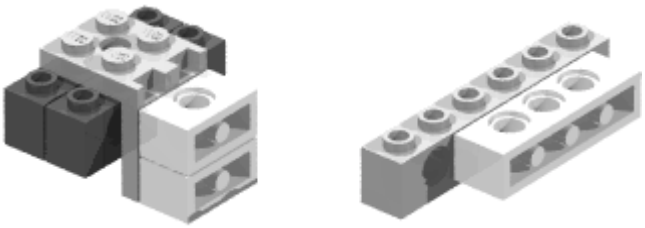


Figure 1-21. Turning 90 degrees. Studs In

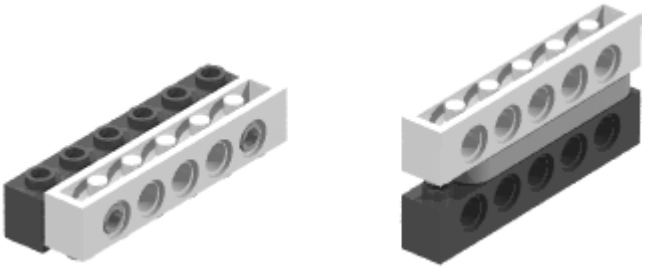


Figure 1-22. Upside Down

Figure 1-23. Extending Beams Using Pins and Plates

2 Gears

Eventually you will want to make your robot move. After all, this is not the FLL Sculpture Competition! The 9 volt motors provide the motive power, but they may not run at the right speed or be powerful enough. It may also be too difficult to position the motors where they can be directly attached to the wheels. All these problems can be solved using gears.

Gears are generally used for one of the following reasons:

1. To transmit torque from one axle to another
2. To increase or decrease the speed of rotation
3. To reverse the direction of rotation
4. To move rotational motion to a different axis
5. To change rotary motion to linear motion
6. To keep the rotation of two axles synchronized

2.1 Spur Gears

A spur gear is used when shafts must rotate in the same plane. In a spur gear the teeth are straight and parallel to the shaft. They are by far the most common type of gears, and they are what most people picture when you mention gears. LEGO includes four different sized spur gears in the Robotics Invention System.

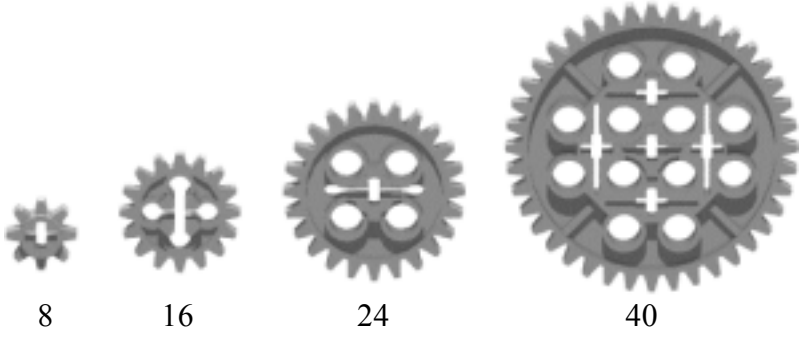


Figure 2-1 Lego Spur Gears

Gears are normally referred to by their type and the number of teeth. Take, for example, an 8 tooth spur gear. Sometimes a kind of shorthand notation is used where ì toothî is replaced with ì tî and the type is not specified for spur gears (because they are so common). For example, a 40 tooth spur gear would be referred to as a 40t gear.

2.1.1 Gear Spacing

Sizes of LEGO spur gears are shown in Table 2-1. It is interesting to note that the ratio of the radii is equal to the ratio of the tooth count (8/24 = 0.5/1.5 = 1/3). This is because all the different sized spur gears have the same sized teeth--even the little 8t gear with its involute profile gear teeth. Having the same sized teeth allows the gears to mesh properly.

Table 2-1. Spur Gear Sizes

Teeth	8	16	24	40
Radius (studs)	0.5	1	1.5	2.5

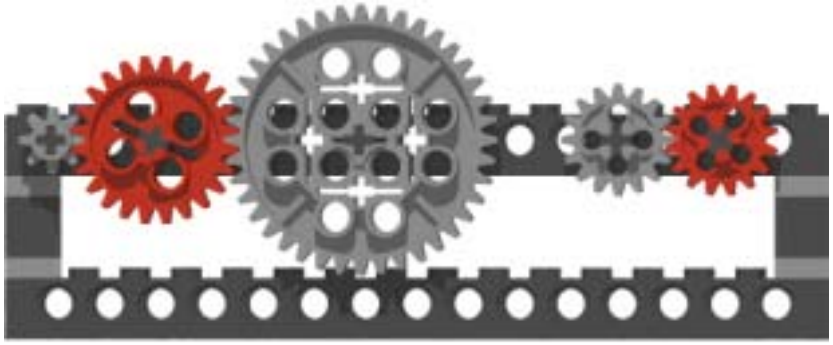
Knowing the radius of a gear and the number of teeth, we can calculate the size of each tooth. This information can be useful to know when using a spur gear with a gear rack.

Circumference = 2 x Pi x Radius
Circumference = Tooth Size x Tooth Count
Size x Count = 2 x Pi x Radius
Size = 2 x Pi x Radius / Count
16 tooth = 2 x Pi x 1 / 16
= Pi / 8 studs
= 0.392 studs or 3.14 mm

Check it out for yourself. The teeth for each spur gear really do evaluate to the same size!

2.1.1.1 Horizontal Gear Spacing

Using the information from Figure 2-2, we can calculate the gear spacing required for proper meshing. The distance between the two axles is equal to the sum of the two radii.



	8t	16t	24t	40t
8t	1.0 studs	1.5 studs	2.0 studs	3.0 studs
16t	1.5 studs	2.0 studs	2.5 studs	3.5 studs
24t	2.0 studs	2.5 studs	3.0 studs	4.0 studs
40t	3.0 studs	3.5 studs	4.0 studs	5.0 studs

Figure 2-2. Stud Gear Spacing

Combinations using eight, twenty-four, and forty tooth gears are straight forward to set up. Sixteen tooth gears are easy to use with other sixteen tooth gears, but require half stud spacing to mesh with 8t, 24t, or 40t gears. A two holed 1 x 2 Technic brick can be used to get half stud spacing.

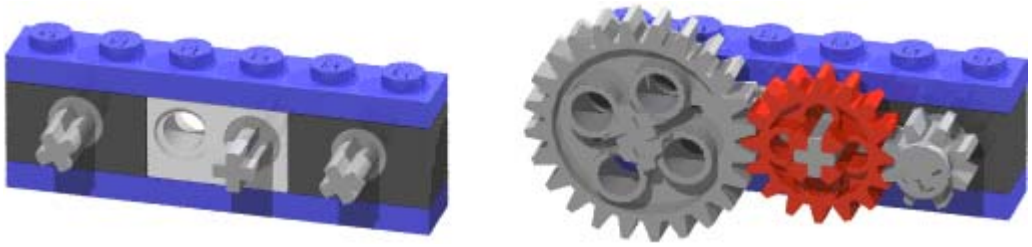
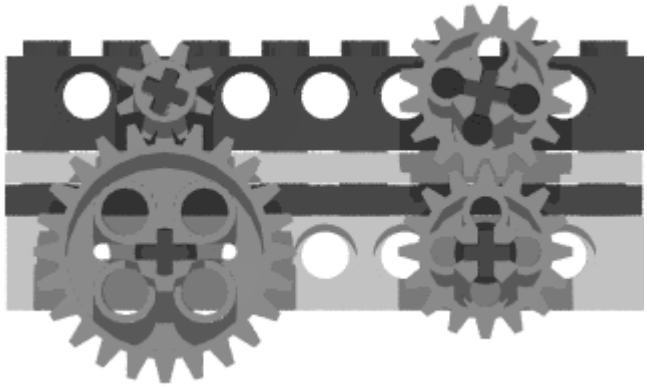


Figure 2-3. Half Stud Spacing Using 2 Holed 1 x 2 Beam

Construction Notes: Gear spacing of two or three studs provides the maximum number of gear combinations. Two stud spacing also works well in vertical layouts.

2.1.1.2 Vertical Spacing

It is difficult to mesh gears laid out on a vertical axis. The gear spacing table shows that proper gear meshing is achieved at half stud intervals starting at 1 stud and going up to 5 studs. For good vertical gear meshing, we need to build a structure using 1.2 stud high beams and 0.4 stud high plates that provides the correct spacing. It works out that 2 and 4 studs spacing are the only solutions.



	8t	16t	24t	40t
8t			2.0 studs	
16t		2.0 studs		
24t	2.0 studs			4.0 studs
40t			4.0 studs	

Figure 2-4. Vertical Gear Spacing

The spacing doesnít have to be perfect for the gears to mesh. Anything within 0.08 studs works fairly well. The 8 tooth and 16 tooth gears almost fit at 1.6 stud spacing (ideal is 1.5 studs), but they donít mesh securely, and it is possible for the gears to slip.

Figure 2-5. Circumventing Vertical Gear Spacing Restrictions

Figure 2-5 shows one way to get around vertical gear spacing restrictions. The middle gear is held in place by an axle pin in the vertical beam. The top and bottom gears are spaced six studs apart allowing them to fit nicely on axles passing through the horizontal beams.

2.1.1.3 Diagonal gear spacing

Diagonal gear spacing is trickier to calculate than perpendicular or vertical gear spacing. As was done to calculate diagonal cross bracing solutions, Pythagorasís theorem for right triangles is used to compute the diagonal (hypotenuse). This value is then compared to the ideal gear spacing information from the table in Figure 2-2.

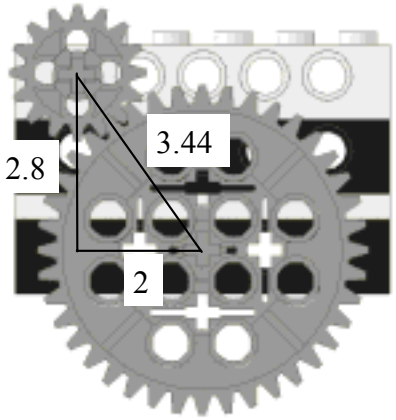


Figure 2-6. Diagonal Gear Spacing

Table 2-2 shows the possible layouts for 8, 16, 24, and 40 tooth spur gears. Vertical dimensions are displayed in the left hand column, horizontal dimensions in the top row. At the intersection of each row and column is the calculated diagonal gear spacing. The entry is left blank if this distance does not match the spacing of any of the available gear combinations. Acceptable gear meshes must be within 0.08 studs of the ideal value. Excessive binding of the gears occurs if the spacing is too small. If the spacing is too great, the gear train will have large amounts of backlash, and slippage may occur.

Table 2-2. Diagonal Gear Spacing

Horizontal Spacing

Vertical Spacing	0	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5
0			1	1.5	2	2.5	3	3.5	4		5
1.2			1.56	1.92							
1.6					2.56	2.97					
2	2	2.06		2.5				4.03		4.92	
2.4		2.45				3.47					
2.8			2.97		3.44						
3.2				3.53		4.06					
3.6								5.02			
4	4	4.03					5.00				
4.4						5.06					
4.8				5.03							

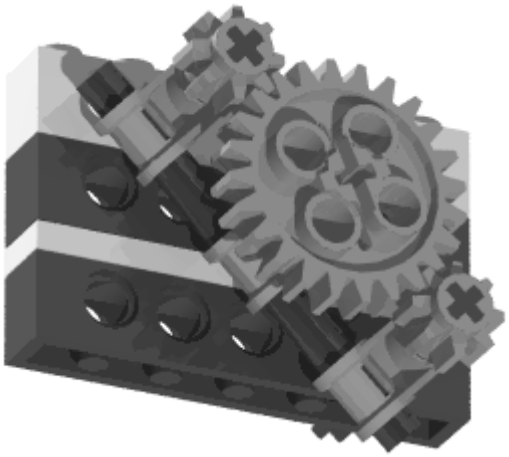


Figure 2-7. Circumventing Diagonal Gear Spacing Restrictions

Figure 2-7 shows one way to get around diagonal gear spacing restrictions. Three perpendicular axle joiners are used to hold the gears in place. The middle 24 tooth gear is attached using an axle pin. The two 8 tooth gears are spaced 4.1 studs apart to fit on axles passing through the horizontal beams. The extra 0.1 studs space is divided evenly between the two gear meshes.

2.1.2 Gear Ratio

Gear ratio is how much the output shaft of a gearbox turns for a given rotation of the input shaft. In Figure 2-8, we have a gearbox consisting of two gears: an 8t gear on the input shaft and a 24t gear on the output shaft. If the 8t gear rotates one full revolution then eight of its teeth would pass through the starting line. Because the two gears are meshed, eight of the 24t gearís teeth would also pass the starting line. Since the teeth are evenly distributed around the circumference of the gear, the 24t gear turns 8/24ths or 1/3 of a revolution.

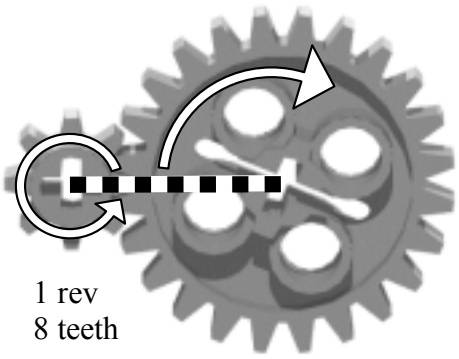


Figure 2-8. 3:1 Gear Ratio

Using the rotation information we can calculate the gear ratio. Following popular convention, it is expressed as a ratio of whole numbers.

$$\textbf{Gear Ratio} = 1 : 1/3$$
$$= 3 : 1$$

The 3:1 gear ratio tells us that the input shaft (attached to the 8t gear) has to complete three full revolutions for the output shaft (attached to the 24t gear) to rotate all the way around just once. Using gears to slow down rate of rotation or decrease the amount of rotation is called gearing down. If we were to switch the 8t and 24t gears around, the output shaft would spin three revolutions for each revolution of the input shaft. This is gearing up, and the gear ratio would be 1:3.

You may have noticed that the gear ratio is the inverse of the ratio of the number of gear teeth. The reason for this is easier to see if we recalculate the gear ratio using an input shaft rotation of only 1 tooth. In the case of the 8t gear driving the 24t gear, the input shaft would turn 1/8 of a revolution and the output shaft 1/24 of a revolution.

$$\textbf{Gear Ratio} = 1/8 : 1/24$$
$$= 24 : 8$$
$$= 3 : 1$$

Using gear tooth counts to directly calculate gear ratios is easier and faster than calculating the rotations first and then using these values to derive the gear ratio.

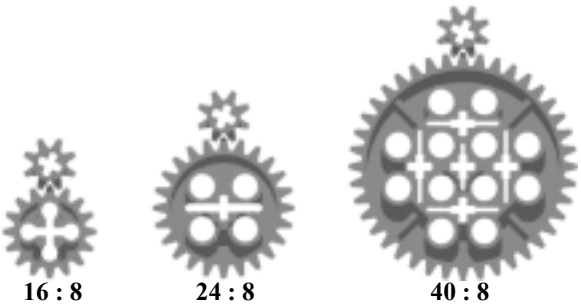


Table 2-3. Gear Ratios for LEGO Spur Gears

Input Shaft or Driving Gear \ Output Shaft or Driven Gear	8t	16t	24t	40t
8t	1:1	2:1	3:1	5:1
16t	1:2	1:1	3:2	5:2
24t	1:3	2:3	1:1	5:3
40t	1:5	2:5	3:5	1:1

Links: Ted Cochran has a nice treatment of gear ratios with Minnesota FLL examples at www.hightechkids.org/fll/coaching/Gears/gears.htm.

2.1.3 Torque

Torque is a force that tends to rotate or turn things. You generate a torque any time you apply a force to the handle of a wrench. This force creates a torque on the nut, which tends to turn the nut. If the nut is too tight, you either pull harder (more force), or get a longer handled wrench (more distance).

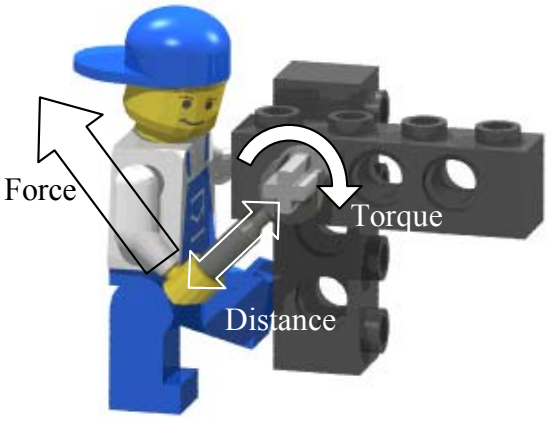


Figure 2-9. Torque = Force x Distance

From the wrench example, we see that torque is a product of force and distance. The units used when measuring torque reflect this. In the U. S. we measure torque in foot-pounds (ft-lbs). Newton-meters (Nm) is the unit of torque in the metric system.

Gears operate by transmitting forces at the teeth of the gear. In Figure 2-10, the 24t gear is generating a force against the teeth of the 40t gear. The force (f) is equal to the torque (t1) applied to the 24t gear divided by the radius (r1). The force transmitted by a gear is inversely proportional to the gearís radius. The larger the radius of the gear, the less force it will generate for a given torque.

The force against the teeth of the 40t gear creates a torque (t2) equal to the force (f) times the radius (r2). The torque created by applying a force to a gear is proportional to the gearís radius. Applying a force to a large gear will create more torque than applying the same force to a small gear.

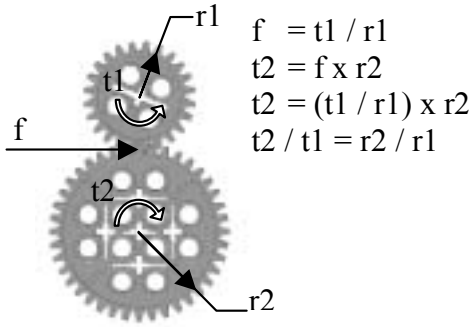


Figure 2-10. The Ratio of the Torques is Equal to the Ratio of the Radii

The torque available at the axle of driven gear (40t) can be expressed as a function of the torque turning the driving gear (24t) and the two radii; t2 = t1 x r2 / r1. A small gear driving a larger gear will amplify torque. There will be more torque available at the shaft of the larger gear than is supplied to the shaft of the smaller gear.

From the discussion earlier, we know that the gear ratio is the inverse of the ratio of the gearsí radii. This allows us to use the gear ratio to calculate the torque amplification of a gear system. Using the gear ratio for the example above:

Gear Ratio = r2 : r1
= r2 / r1
= 5 / 3
t2 = t1 x r2 / r1
= t1 x Gear Ratio
= t1 x 5 / 3

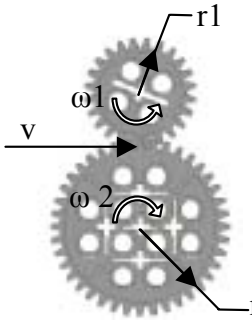
The Technic gear motor supplied with the RIS kit can generate a torque of about 9Ncm with fresh batteries. In the example above, how much torque would be available at the shaft of the 40t gear if we attached the motor directly to the shaft of the 24t gear?

t2 = t1 x 5 / 3
= 9 Ncm x 5 / 3
= 15 Ncm

2.1.4 Speed

When using simple machines like gears (or any kind of machine for that matter) you never get something for nothing. In our prior example, we used gears to increase torque. What we traded to get the torque increase was speed. The output shaft may turn stronger, but it also turns slower.

If we measure the angles in radians (1 radian = 180 degrees / Pi = 1 revolution / (2 x Pi)), the tooth velocity of a gear (v) is equal to the angular velocity (ω) times the radius (r). There is a proportional relationship between the radius and the tooth velocity. At a given angular velocity, the teeth of a larger gear will travel faster than the teeth of a smaller gear.



$v = \omega 1 \times r1$

$\omega 2 = v / r2$

$\omega 2 = (\omega 1 \times r1) / r2$

$= \omega 1 \times r1 / r2$

$\omega 2 / \omega 1 = r1 / r2$

Figure 2-11. Ratio of Angular Velocities is Equal to Inverse Ratio of the Radii

When two gears are meshed, the tooth velocity for each gear is the same. In the example above, what is the angular velocity of the 40t gear if the 24t gear is spinning at 180 rpm?

r1 = 1.5 studs = 12 mm
r2 = 2.5 studs = 20 mm
ω1 = 180 rpm
= 180 revolutions / minute x 1 minute / 60 seconds
= 3 revolutions / second
= 3 revolutions / second x 2 Pi radians / revolution
= 6 Pi radians / second
= 18.85 radians / second
v = ω1 x r1
= 18.85 radians / second x 12mm / radian
= 226.19 mm / second
ω2 = v / r2
= (226.19 mm /second) / (20 mm / radian)
= 11.31 radians / second
= 1.8 revolutions / second
= 108 rpm

The larger gear spins more slowly than the smaller gear. The ratio of the angular velocities is equal to the inverse of the ratio of the radii. Knowing this we can calculate the angular velocity using the radius ratio directly.

ω2 = ω1 x r1 / r2
= 180 rpm x 1.5 studs / 2.5 studs
= 108 rpm

Just as with torque, the gear ratio can be used in place of the ratio of the radii.

Gear Ratio = r2 : r1
= r2 / r1
= 5 / 3
ω2 = ω1 x r1 / r2
= ω1 x (1 / Gear Ratio)
= ω1 / Gear Ratio
= 180 rpm / (5 / 3)
= 180 rpm x 3 / 5
= 108 rpm

Links: How Stuff Works, one of the best sites on the web, has a wonderful article about gears, gear ratios, torque, and speed. Check it out at "www.howstuffworks.lycozone.com".

2.1.5 Gear Trains

If a number of gears are cascaded so that several meshes relate an input to an output, a gear train is formed. Using four 40t and four 8t gears, it is possible to create a gear ratio of 625:1.

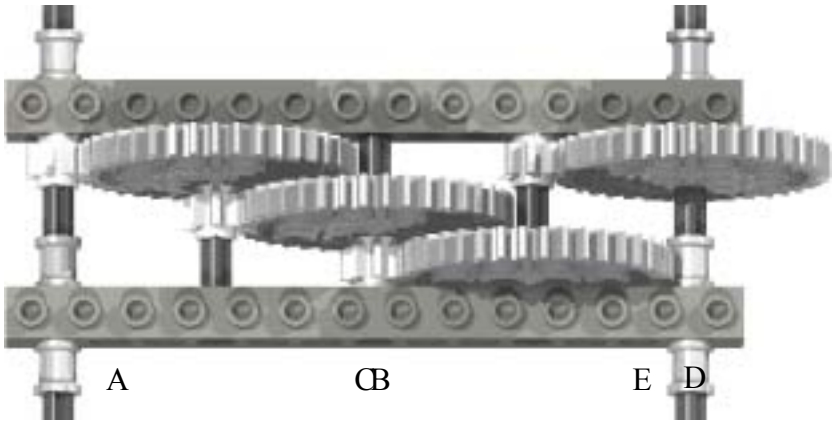


Figure 2-12. Multi-stage Gear Train

The gear train above consists of four stages, each having a 5:1 gear ratio. To calculate the overall gear reduction, we start with the gear ratio between A and B, multiply that times the ratio between B and C, and so on until we get to E.

A→B = 5:1
B→C = 5:1
C→D = 5:1
D→E = 5:1
A→E = A→B x B→C x C→D x D→E
= 5:1 x 5:1 x 5:1 x 5:1
= 5 x 5 x 5 x 5:1 x 1 x 1 x 1
= 625:1

The 9v geared motor can generate about 0.06 ft-lbs of torque with fresh batteries. If we hooked one up to shaft A and turned it on, the torque amplifications should produce 37.5 ft-lbs of torque at shaft E. Thatís enough torque to tighten the lug nuts on my carís wheels. Due to friction in the gear train, the actual available torque will be much less than 37.5 ft-lbs, but it will be strong enough to snap axles and break gear teeth. Be careful when using lots of gear reduction.

As mentioned before, any increase in torque is accompanied by a corresponding decrease in rotation speed. If we took our motor, hooked it up to shaft A and got it spinning at 200 revolutions per minute (rpm), shaft E would spin at 0.32 rpm or about 1 revolution every 3 minutes. If instead, we attached the motor to shaft E, then shaft A should spin at 125,000 rpm. That would have the teeth of the last 40t gear traveling at 585 mph! Fortunately, the large amount of friction in the gear train prevents the weak 9v motor from generating such dangerous speeds. When gearing up to increase speed, torque amplification magnifies the friction forces of the later stages. With a gear ratio of 1:625, it is unlikely that the motor is powerful enough to spin the gears at all.

Question: For the example above, what would the gear ratio be if the 8 tooth gears were replaced with 24 tooth gears?

2.1.5.1 Idler Gear

The 24 tooth gear in Figure 2-13 is an idler gear. An idler gear does not affect the gear ratio of a gear train. The gear ratio for A→C is the same as it would be if the 24 tooth gear were left out. Idler gears are quite common in machines where they are used to connect distant axles. They are also used to change the direction of rotation of the output shaft.

A→B = 3:1
B→C = 1:3
A→C = A→B x B→C
= 3:1 x 1:3
= 3 x 1:1 x 3
= 1:1

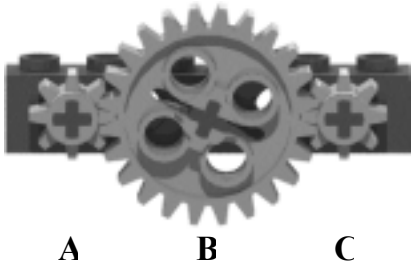


Figure 2-13. Idler Gear

2.1.6 Clutch Gear

The funny looking white 24 tooth spur gear with the writing on its face is the clutch gear. The clutch gear is special in that the gear teeth are able to rotate about the shaft. It has an internal clutch mechanism that starts to slip when its maximum rated torque is exceeded. The clutch gear is used to limit the torque of a geared system, saving motors and preventing your robot from tearing itself apart.

The clutch gear has ì2.5˙5 Ncmî stamped on its face. This is the torque rating of the clutch. Ncm stands for Newton Centimeter, a unit of torque (torque is a product of force and distance, centimeter is a unit of distance, and Newton is a unit of force). The clutch gear can transmit a maximum torque of from 2.5 to 5 Ncm (0.018 to 0.037 ft lbs or 0.22 to 0.44 in lbs).

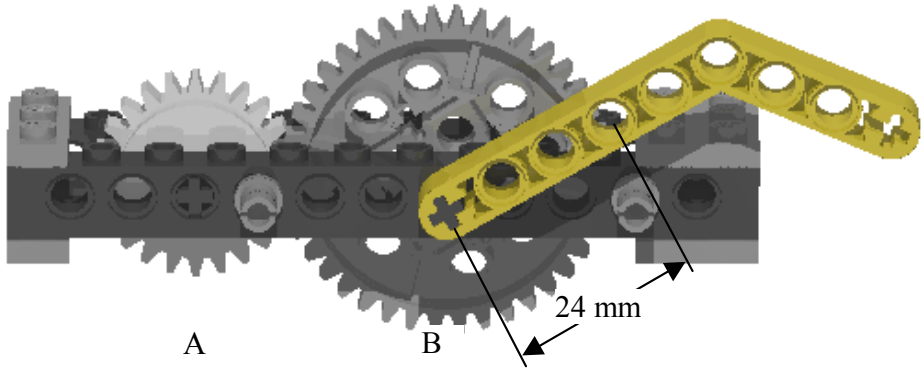


Figure 2-14. Using Clutch Gear to Limit Forces

In Figure 2-14, the clutch gear is used to limit the force of the lift arm pressing against the connector peg stops. Without the clutch gear, we would run the risk of stalling a motor or damaging the assembly. Using the information we have about the clutch gear, gear ratios, and distance, itís possible to calculate the maximum force the lift arm can generate to push against the connector peg.

A max torque = 5 Ncm
Gear ratio A:B = 5:3
B max torque = A torque x Gear Ratio
= 5 Ncm x 5 / 3
= 8.33 Ncm

Distance B to Pin = 3 studs
= 24 mm or 2.4 cm
Max force at Pin = B torque / Distance
= 8.33 Ncm / 2.4 cm
= 3.47 N or 0.78 lbs

2.2 Crown Gear

The crown gear has teeth that are raised on one side and rounded off on the other. This gives it a crown-like appearance. The crown gear is used when the shafts to be turned meet at an angle--usually a right angle. The crown gear can be meshed to spur gears and worm gears, but it doesnít mesh well with other crown gears. The crown gear can also be used in place of a 24 tooth spur gear.

Figure 2-15. Crown Gear

Treat the crown gear just like the 24t spur gear when computing gear reduction. For the gearbox above, if the 8t gear is attached to the input shaft, and the 24t spur gear to the output shaft, the gear ratio is:

Gear Ratio = 24:8 x 24:24
= 3:1 x 1:1
= 3:1

2.3 Bevel Gear

The bevel gear has teeth that slope along one surface of the disc. It is used when the shafts to be turned meet at an angle--usually a right angle. It has less friction than the crown gear, but it can only mesh with another bevel gear. LEGO produces 12 tooth, 14 tooth, and 20 tooth bevel gears. Unfortunately, only the 12 tooth bevel gear is included in the RIS kit.

Figure 2-16. Bevel Gear

The old 14 tooth bevel gears have a problem with their thin faceted teeth breaking. The 12 tooth bevel gear has a circular backing plate that strengthens the gear teeth to prevent this from happening. The backing plate gives the gear a smooth circular outline and allows it to be used as a very small wheel.

Figure 2-17. A Small Wheel Built from Two 12t Bevel Gears

Question: A wheel made out of bevel gears would not have much traction. For what applications could this be a benefit?

2.4 Worm Gear

A worm gear is a screw that usually turns along a spur gear. Motion is transmitted between shafts that are at right angles. If you want to create a high gear ratio, nothing beats a worm gear. Each time the shaft spins one revolution, the spur gear moves one tooth forward. If the spur gear has 24 teeth, you have a 24:1 gear ratio in a very small package.

Figure 2-18. Worm Gear

The efficiency of a worm gear system is much lower than that of normal meshes because the worm works primarily by sliding, thus increasing frictional losses. This has an unusual side effect in that the worm gear is asymmetric and self-locking. You can turn the input shaft to drive the output shaft, but you cannot turn the output shaft to drive the input shaft.

Figure 2-19. Using the Worm Gear's Self Locking Feature

The mechanism in Figure 2-19 makes use of the worm gearís self-locking feature to hold the bucket in place. The axle through the worm gear can be turned to raise or lower the bucket. Once in place, no torque is required to maintain the position.

The worm gear can also be used to implement a lead screw. Lead screws convert rotary motion to linear motion. A lead screw consists of a threaded rod and a captured nut. Spinning the rod causes the nut to move along the length of the rod. Lead screws are used in many devices. One of the most visible examples is the Genie garage door opener.

Figure 2-20. LEGO Lead Screw

Figure 2-20 is a LEGO implementation of a lead screw. The threaded rod is constructed of multiple worm gears on the center axle. The half bushings on the outer axles make up the captured nut. Spinning the center axle causes the outer axles to move in or out.

Construction Note: Be careful when constructing a threaded rod out of worm gears. You can put two worm gears on an axle in four different ways, but only one results in a continuous thread.

Good Bad

2.4.1 Directional Transmission

One of the cleverest uses I have seen for a worm gear is the directional transmission. A directional transmission lets you use one output port to perform two functions. It has one input shaft and two output shafts. A worm gear slides along the input shaft and engages spur gears on the output shafts. Which gear is engaged is dependent upon the rotation direction of the input shaft.

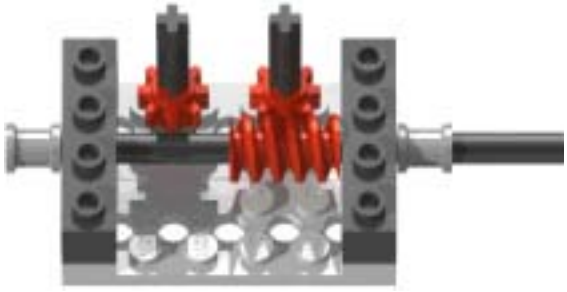


Figure 2-21. Directional Transmission

The directional transmission works because of friction. When you spin the input shaft, the worm gear exerts a force against one of the spur gears. The spur gear resists turning because of friction and pushes back against the worm gear. If free to do so, the worm gear slides along the input shaft, engaging the other spur gear and eventually running into the support beam. Now the force required to slide the worm gear exceeds the force required to turn the spur gear, and the spur gear begins turning. Reversing rotation of the input shaft causes the whole sequence to occur again but in the opposite direction.

2.5 Differential

A differential is a device that takes a torque applied to its housing and evenly distributes it to two output shafts, allowing each output to spin at a different speed. Differentials are found on all modern cars and trucks. All-wheel drive cars like the Audi Quattro can have three differentials; one between the front tires, one between the rear tires, and one between the front and rear differentials. LEGO provides a differential with the RIS kit.

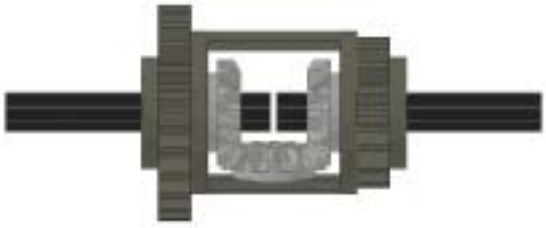


Figure 2-22. LEGO Differential

Car wheels spin at different speeds when turning. Each wheel travels a different distance through the turn with the inside wheels traveling a shorter distance than the outside wheels. Since **speed = distance / time**, the wheels that travel a shorter distance travel at a lower speed.

This is not a problem for wheels that can spin independently. But in most vehicles, the driven wheels are linked together so that they can be powered by a single motor. Without

a differential, the wheels would be locked together, forced to spin at the same speed. This would make turning difficult. For the car to be able to turn, one wheel would have to slip, which can require great force.

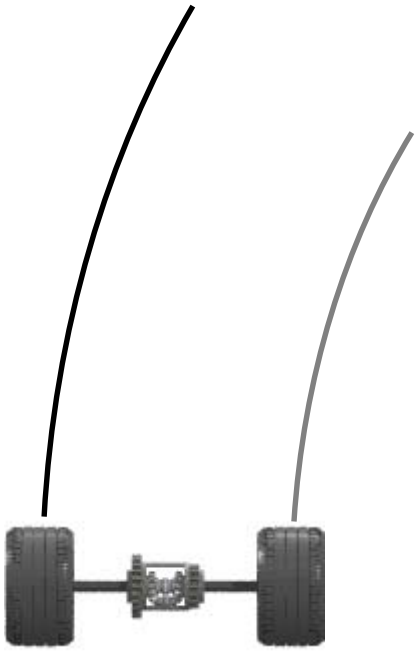


Figure 2-23. During Turns the Wheels Cover Different Distances

The differential is a mechanical calculator. It computes the average rotation speed of the two input axles and spins the differential housing at this rate. This is an interesting property that can be used in many ways. In Figure 2-24, the differential is used to calculate the average rotation speed of the two wheels. Attaching a rotation sensor to the differential housing would provide an accurate measure of travel distance.

Wheel 1	Wheel 2	Housing
100 rpm	100 rpm	(100 + 100) / 2 = 100 rpm
70 rpm	50 rpm	(70 + 50) / 2 = 60 rpm

80 rpm	-80 rpm	(80 ñ 80) / 2 = 0 rpm

Figure 2-24. Using Differential to Calculate Average Rotation

The platform below uses the differential to measure the difference in rotation speed between the two axles. Notice the extra gear between the differential and the wheel on the right side. It reverses the direction of rotation for the right differential shaft. A rotation sensor could be used here to measure turning.

Wheel 1	Wheel 2	Housing
100 rpm	100 rpm	(100 - 100) / 2 = 0 rpm
70 rpm	50 rpm	(70 - 50) / 2 = 10 rpm
80 rpm	-80 rpm	(80 + 80) / 2 = 80 rpm

Figure 2-25. Using a Differential to Calculate Difference in Rotation

2.5.1 Ratchet Splitter

An intriguing use for the differential is as part of a ratchet splitter. A ratchet splitter lets you perform two different functions with a single motor, just like the directional transmission. But instead of using friction to switch gears, the ratchet splitter uses a ratcheting mechanism to prevent an axle from spinning in both directions.

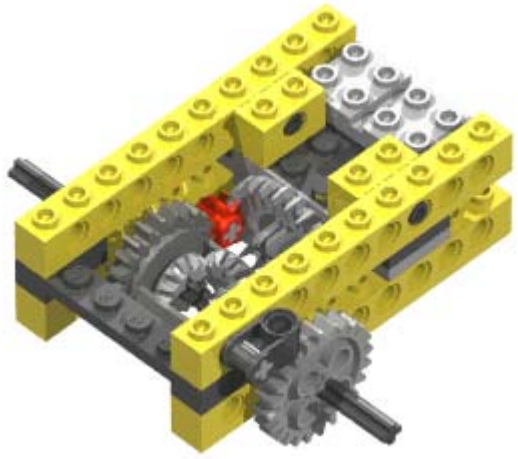


Figure 2-26. Ratchet Splitter

The assembly in Figure 2-26 is part of a differential drive car I built that used a single motor to provide both locomotion and steering. The perpendicular axle joiner and 24t gear is a ratcheting mechanism that prevents the right axle from rotating counter clockwise. When the motor turns clockwise, the left and right axles turn in the same direction and propel the vehicle forward. When the motor turns counter clockwise, the ratchet locks the right wheel, and the vehicle turns to the right as it backs up.

Question: What would happen if a second ratchet was added to prevent the left axle from turning clockwise?

2.6 Gear Rack

The gear rack looks like a spur gear laid out flat. It is usually used in conjunction with a spur gear (which is referred to as the pinion). Rack and pinion gears are used to convert rotation into linear motion. An example of this is the steering system on many cars. The steering wheel rotates a gear that engages the rack. As the gear turns, it slides the rack either to the right or left depending on which way you turn the steering wheel. The motion is transmitted via linkages to the front wheels causing them to pivot.

Figure 2-27. LEGO Gear Rack and Pinion

One of the trickier problems encountered when using the gear rack is how to provide a smooth surface upon which the rack can slide. Most LEGO car models solve this problem with tiles, which are plates without studs on top. Figure 2-27 uses two 1 x 5 lift arms to provide a smooth surface. An upside down Technic beam is another commonly used solution.

2.7 Pulleys

A pulley is a wheel with a groove about its diameter. The groove, called the race, accepts a belt which attaches the pulley to other pulleys. As the pulley rotates, friction forces pull on the belt putting it in tension. The belt transmits the force to the other pulley causing it to rotate.

Figure 2-28. Pulleys and Belts

LEGO includes four sizes of pulleys in the RIS--the half bushing, the small pulley, the pulley wheel, and the large pulley. Pulleys are connected using belts that rest in the pulleysí races. LEGO belts are color coded: small (white), medium (blue), and large (yellow). The black rubber bands can be used as belts, but they are more elastic than the LEGO belts, and their rectangular cross section doesnít fit well in a pulleyís race.

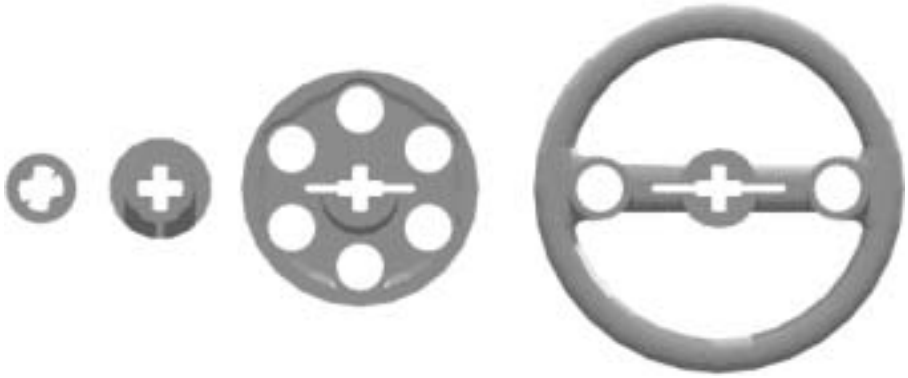


Figure 2-29. Lego Pulleys

With four different sized pulleys, it is possible to ì gearî up and ì gearî down. The reduction ratio with pulleys is determined by the ratio between their diameters. The trick is determining exactly where to measure the diameter. Table 2-4 shows pulley diameters measured at the bottom of the race. The resulting reduction ratios are close to the values determined experimentally in Table 2-5.

Table 2-4. Measured pulley diameters

Diameter	Half Bushing	Small Pulley	Medium Pulley	Large Pulley
Millimeters	5.7	8.7	22	34.5
Studs	0.71	1.09	2.75	4.3

Table 2-5. Pulley Ratios

	Half Bushing	Small Pulley	Medium Pulley	Large Pulley
Half Bushing	1:1	3:2	7:2	6:1
Small Pulley	2:3	1:1	7:3	4:1
Medium Pulley	2:7	3:7	1:1	5:3
Large Pulley	1:6	1:4	3:5	1:1

2.7.1 Torque

Pulleys may be used in place of gears in many applications. Since there are no teeth to mesh, placement is much more forgiving. But because it has no teeth, a pulley cannot be used to transmit high torques. The belt will slip first. Determining when and how much the belt will slip is difficult. It depends on many factors such as the size of the pulleys, the tension of the belt, and the friction between the pulley and the belt.

You can increase the torque capacity of a pulley system by increasing the reduction ratio or by increasing the friction between the pulleys and the belt. Increasing the reduction ratio may not be desirable because it has an associated decrease in speed. Friction can be increased either by using a tighter belt or by increasing the contact area between the pulleys and the belt.

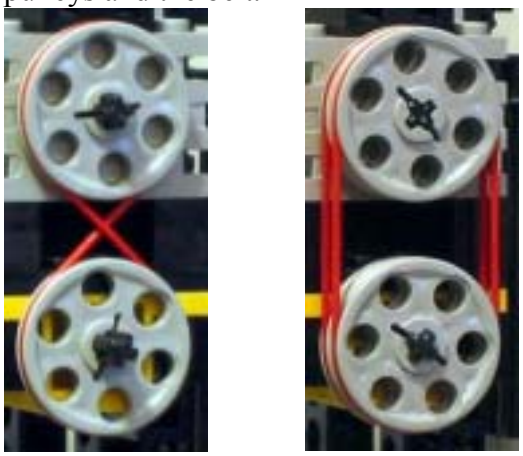


Figure 2-30. Two ways to Increase Torque Capacity

Question: How else can we increase the friction between the pulleys and the belt?

Some enterprising robot designers use belt slippage as a torque limiting device, similar to the clutch gear. Because slippage is difficult to predict, it is best to use experimentation to find the belt and pulley combination that will slip at the desired torque.

Figure 2-31. Using Pulleys to Limit Torque

Construction Notes: The tank tread and sprocket can be used like a pulley and belt system. The tread's teeth prevent slipping even if a large torque is applied to the sprocket.

2.8 Reinforcing gear trains

The components of the gear train are likely to experience the largest forces of anything in your robot. The motor will stall and wheels will slip before forces get too out of hand. But with enough gear reduction, it is easy to generate forces that will tear a gearbox apart.

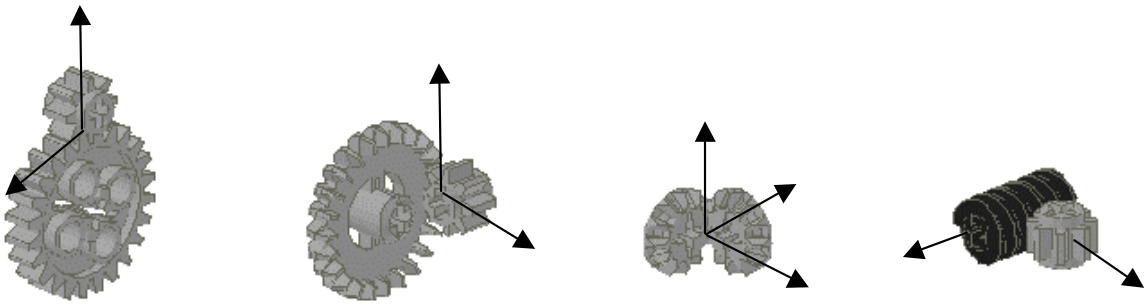


Figure 2-32. Forces on gears

Different types of gears require different kinds of support. In spur gears, forces are applied perpendicular to the axles. Unless the axles are properly supported, they will be pushed apart when high torques are applied. Supporting the axles is not a problem in horizontal layouts, but vertical and diagonal layouts may require additional bracing.

Crown and bevel gears have forces directed perpendicular and parallel to their axles. As with spur gears, the supporting structure will probably require bracing to hold the axles in place. This can be difficult at times because the supports are perpendicular to each other. Crown and bevel gears also require a solid backing to prevent them from sliding on their axles.

The most difficult problem with the worm gear is locking the worm screw in place on its axle. This is exacerbated by the gearís odd length (itís 15.5mm long instead of the 16mm

required for a perfect fit). When the worm gear is used with a spur gear, the spur gearís axle must be well supported.

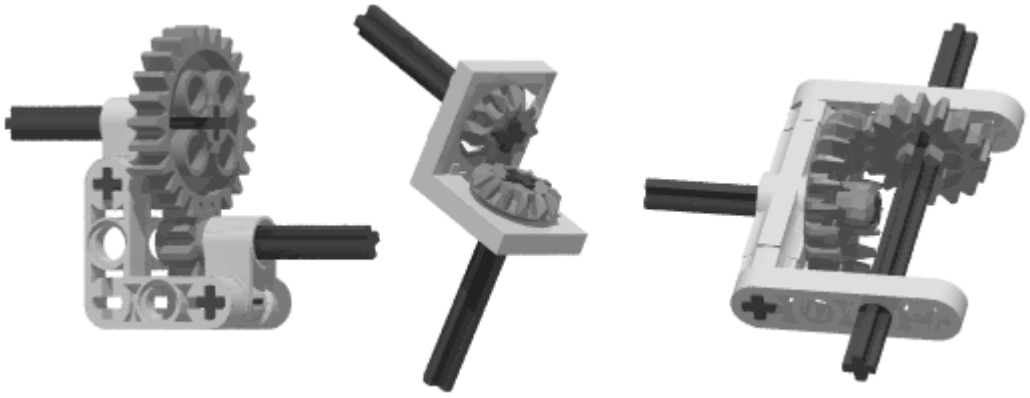


Figure 2-33. Gearboxes Donít Have to be Big to be Strong

2.9 Backlash

The backlash of a gear train is the amount the input shaft can rotate without moving the output shaft. Backlash is caused by the gears not meshing perfectly (see Figure 2-34). In this example, when gear A reverses rotation, the tooth on gear B goes from being loaded on the left side to being loaded on the right side. Because of the gap between the teeth, A will be able to rotate a small amount before B notices the change in direction.

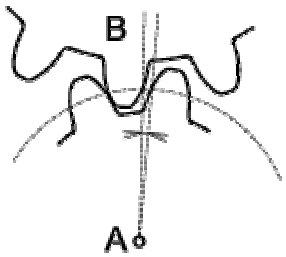


Figure 2-34. Backlash is caused by poor gear meshing

Backlash introduces discontinuity, uncertainty, and impact in mechanical systems. This makes accurate control difficult. Positioning accuracy is also compromised due to backlash. A large amount of backlash makes a robot feel sloppy and gives an overall impression of poor engineering (you want to impress those judges).

In industry, backlash is reduced by using specially mated gears that are designed to mesh perfectly. LEGO gears are designed to work with a wide variety of gears of different sizes and designs. This limits how perfectly they can fit together. Luckily, there are other ways to minimize backlash and its effects.

Reducing Backlash:

1. Place rotation sensors near the output shaft. This minimizes the effect backlash has on the sensor readings. Unfortunately, it also limits your ability to increase sensor resolution by gearing up.
2. Use large gears. Backlash is less noticeable in larger gears.
3. Minimize the number of gears in a gear train. The larger the number of meshes, the greater the backlash.
4. Backlash increases when you gear up and decreases when you gear down.
5. When laying out gears diagonally, keep gear spacing near the ideal values. Backlash increases if the gears are too far apart.
6. Be careful when using the worm gear. The worm gearís odd size (1.94 studs long) makes it difficult to support. Multiple half bushings can be used to hold the worm gear in place for lightly loaded applications.

An interesting alternative is to prevent backlash from occurring. You can take the play out of a gear train by applying a torque to the output shaft, trading some input torque for more precision. If the torque is great enough, it prevents backlash from happening when changing direction. In Figure 2-35, the rubber band is supplying the force used to preload the gear train.

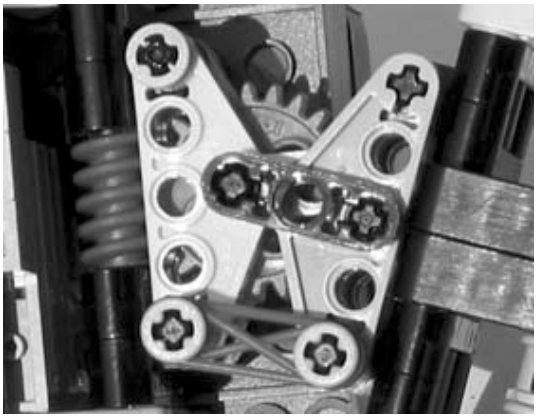


Figure 2-35. Preloading a gear train with a rubber band

Some high precision machines use a split gear to minimize backlash. A split gear looks like a spur gear that has been cut through the middle about its circumference. Springs are placed in channels between two gears. The springs push the gears apart radially. When meshed to another gear, the spring is compressed providing a preload force.

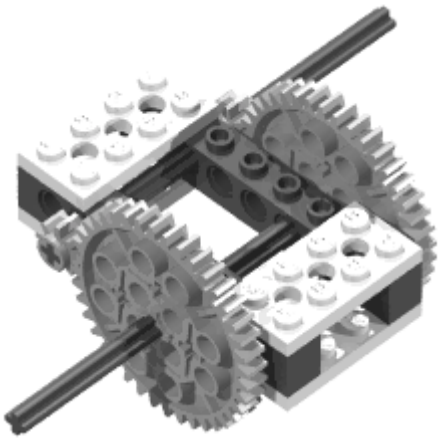


Figure 2-36. Split gear made from LEGO

Figure 2-36 shows a split gear made out of LEGO parts. It uses the axles instead of springs to provide the preload force. When assembling, the 40t gears are twisted one tooth relative to each other before meshing them to the 8t gears.

3 Wheels

FLL challenges involved moving around and picking up things, moving around and dropping things, moving around and triggering things. All this moving around requires some sort of mobile platform, and most of the mobile platforms use wheels.

Wheels support the weight of the robot and transmit the power of the motors to the ground. What wheels you use will affect your robotís speed, power, accuracy, and ability to handle variations in terrain. Wheel choices will have a profound effect on your robotís success or failure.

3.1 Sizes

LEGO has supplied a wide variety of wheels and tires in the RIS kit. There are three sizes of solid rubber tires which all fit on the same plastic wheel and three sizes of balloon tires that fit on differently sized hubs. Dimensions in millimeters are stamped on the sidewall of the balloon tires. Dimensions for the solid tires are not available, therefore, approximate dimensions are supplied in the figure below.

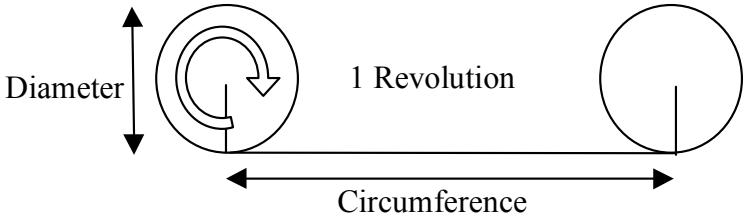
Figure 3-1. Lego Wheels and Tires

The diameter of the wheels chosen will, in a large part, determine how fast and powerful your robot is. Large wheels will make a robot run faster but decrease its towing capacity. Small wheels will give you more power but with a corresponding decrease in speed.

3.1.1 Speed

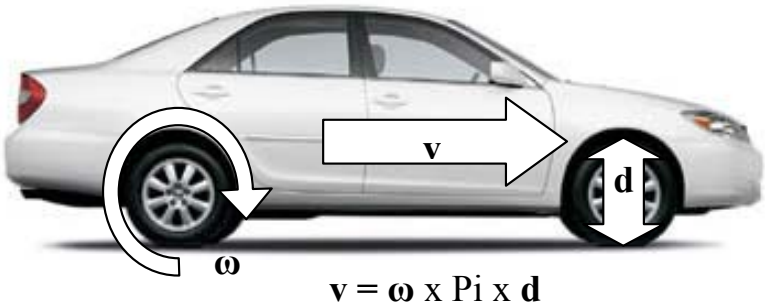
As you drive down the highway the speed your car travels is dependent upon the rotational speed of the engine, the gear ratio of the transmission, and the diameter of the tires. The engine speed and gear ratio define how fast the drive wheels are spinning

(their angular velocity). The angular velocity can then be converted to linear velocity using the equation for the circumference of a circle (distance around the circle).



Circumference = Pi x Diameter

I am driving down the highway in my trusty Camry with the broken speedometer cable when I see a police car with its lights on. A quick look at the instruments shows the engine is at 800 rpm and that I am in third gear. Am I going to get a ticket?



v = ω x Pi x **d**

I know (because I diligently read the owners manual) that the gear ratio for my automatic transmission is 1:1 in third gear. I also know that the tires are 24.6î in diameter. So what is the speed?

ω = Engine RPM x Gear Ratio
= 800 rpm x 1:1
= 800 rpm
v = ω x Pi x d
= 800 rpm x 3.14 x 24.6 inches
= 61826 inches per minute
= 58.5 mph ← **I think I'm OK**

The same equations can be used to calculate the expected travel speed of your robot. For example, letís use the robot shown in Figure 3-2. It has one 9v motor that is geared up by a factor of 3:1 (a 24 tooth crown gear driving an 8 tooth spur gear). It uses the big 81.6 x 15 balloon tires to make it go really fast. The motor should be able to reach 300 rpm for a lightly loaded structure like this. What is the expected speed?

ω = Motor RPM x Gear Ratio
= 300 rpm x 3:1
= 900 rpm
v = ω x Pi x d

= 900 rpm x 3.14 x 81.6mm
= 230,601 mm per minute or 8.7 mph ← **Wow!!!**

Figure 3-2. A Very Fast Tractor

Question: For the example above, what would the travel speed be if we changed the gear ratio by putting the 8 tooth gear on the motor and the 24 tooth crown gear on the axle?

3.1.2 Force

In our discussion of gears, we saw that there is a relationship between force, torque, and radius--the same relationships hold true for wheels. When you use big wheels to increase speed, you have to give up something, and that something is force. A robot with big wheels cannot pull as much as a robot with small wheels.

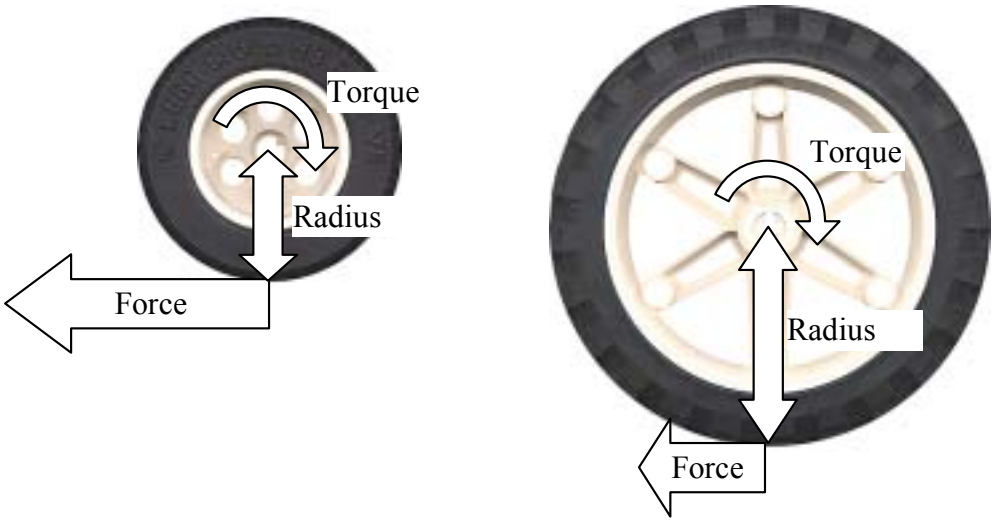


Figure 3-3. Force = Torque / Radius

The relationship between force, distance, and torque works against us when it comes to wheels. Large wheels have a bigger radius (lever length) and will generate less force for a given torque than small wheels would. Less force means less acceleration. The car will go faster, but it will take longer to reach that speed. Using 9 Ncm as an estimate for the available motor torque, letís determine the wheel forces for the vehicle in Figure 3-2.

Torque = Motor Torque x Gear Ratio
= 9 Ncm x 1:3
= 3 Ncm

Radius = Diameter / 2
= 81.6 mm / 2
= 40.8 mm or 4.08 cm

Force = Wheel Torque / Radius
= 3 Ncm / 4.08 cm
= 0.735 N
= 0.1653 lbs or 2.64 oz ← Weak!!!

For comparison, if we used the small solid wheels (24 mm), the tractor would travel at 2.5 mph with the wheels supplying up to 9 oz of force. Switching the gear ratio from 3:1 to 1:3 would have the tractor going only º mph, but it would be capable of delivering a whopping 5 lbs of force. Itís unlikely that the tires would have enough traction to transmit this much force to the ground, consequently, some slipping would result.

Question: If big wheels are less efficient for towing, why do tractors have big wheels?

3.2 Treads

Tracked robots have been very popular in FLL competitions. They are easy to build, and the kids think they look cool. The RIS kit comes with two tank treads and four sprocket wheels. The sprocket wheels have a round axle hole allowing them to spin freely. Driving the tread requires some additional mechanism to lock the sprocket wheels to their axle. This is usually done with a 16 tooth spur gear.

Figure 3-4. Tracked Robot

Treads have good traction in rough terrain and, if well supported, can cross small ravines that would stop a wheeled vehicle. Tracked robots tend to be wide and low to the ground, giving them excellent stability. They are also very agile. A tracked robot can turn in place by spinning its treads in opposite directions.

Unfortunately, treads have a number of disadvantages. They have fairly poor traction on smooth surfaces. The slipping treads make it difficult to use dead reckoning navigation. Also there is a lot of power loss due to the continuous bending and straightening of the treads as they roll around the sprocket wheels.

Figure 3-5. Mario Ferrari's Johnny 5[4]

[4] From www.marioferrari.org/lego.html

Only one size track is included in the RIS kit. For the track not to slip, the spacing between the sprockets must be large enough to maintain a light tension (about 12 studs). The wheelbase can be shortened if additional sprockets are used to change the shape of the track. In the example above, two pulley wheels make up the additional sprocket that gives Johnny 5 his distinctive drive train.

3.3 Balance

Proper balance is very important in robot design. For your robot to move in a predictable and repeatable manner, all wheels must be in contact with the ground at all times, and the weight carried by each wheel must be consistent. Improper balance will result in a robot that tips over easily or lifts its wheels when accelerating or turning.

Balance is dependent upon two factors: wheel base and center of gravity. Center of gravity (CG) is the point within your robot where there are equal amounts of mass above and below, equal amounts to the left and right, and equal amounts fore and aft. For stability calculations, we pretend that all the mass of your robot is concentrated in this tiny spot. The wheel base is the region outlined when you play connect the dots with the points where your robotís wheels touch the ground.

Figure 3-6. Wheelbase

To maintain balance, the CG must remain well inside the wheelbase. If it is outside the wheelbase, the robot will tip over. The closer the CG is to the center of the wheelbase, the more stable the robot becomes.

3.3.1 Finding the Center Of Gravity

Determining your robotís CG is an informative and educational activity. There are many ways this can be done, ranging from the dry and boring (summing the moment of inertia for each component and dividing by the total mass) to the slightly frightening (hanging the robot from a string). A fairly safe and easy way to locate the CG is the balance method.

To determine the CG using the balance method you need to locate the balance point on the lateral, longitudinal, and vertical axes. Each balance point defines a plane upon which the CG resides. The CG is located at the intersection of the three planes.

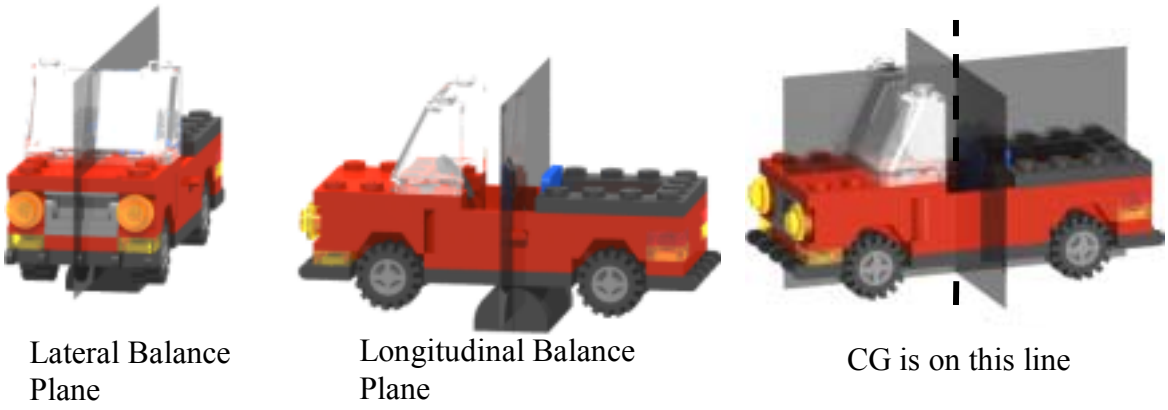


Figure 3-7. Finding CG Using Balance Method

To find the balance points, you need a fulcrum that will support the weight of the robot while still allowing it to tip easily. A 2 x 4 curved top brick is the fulcrum in the example above. Place the robot on the fulcrum such that the fulcrum is parallel to the balance plane you are trying to locate. Slowly adjust the position of the fulcrum until the robot balances. This is the balance point. Repeat for the other two axes to find the CG.

Finding the vertical component of the CG can be done the same way, but sometimes your robotís design will allow this. If I canít balance the front, rear, or sides of the robot on a fulcrum, I modify the balance method to use a part of the robot as the fulcrum. Carefully tilt the robot up until it feels like it is balanced. Now use a carpenters square (a book or tissue box will work as well) to project a plane up vertically from the fulcrum. The CG is the point where this plane intersects the lateral and longitudinal balance planes.

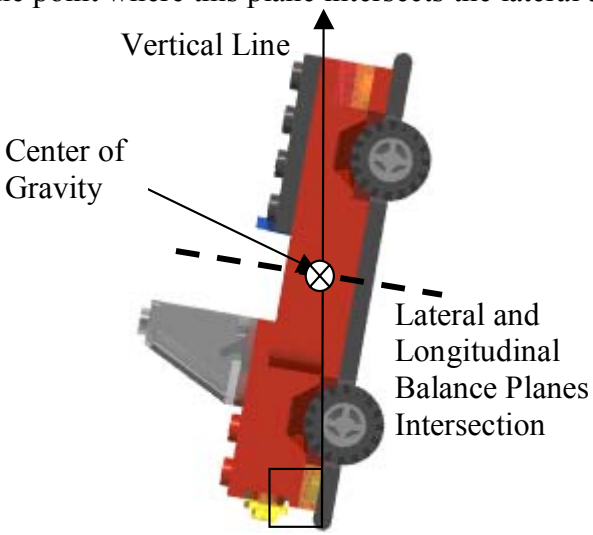


Figure 3-8. Modified Balance Method

3.3.2 Inertia

Determining the vertical component of the center of gravity is more difficult than finding the left/right center or the fore/aft center. So why bother doing it? After all, if you know

that the CG is near the center of the wheelbase, then the robot is guaranteed to be stable, right? Well, not always. Figure 3-9 shows how climbing an incline moves the effective CG. The robot with the lower CG is still stable, but climbing the incline moves the CG behind the rear wheels on the robot with the higher CG, causing it to tip over backwards.

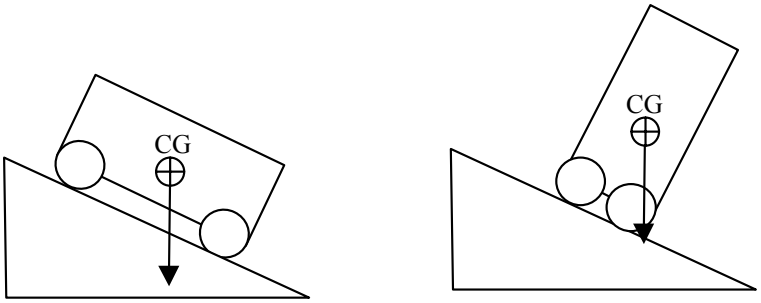


Figure 3-9. An Incline Moves the Effective CG

A more important reason for knowing the vertical CG is to be able to predict the effects that acceleration will have on your robot. The widely publicized (and over hyped) problems with the Suzuki Sidekick and other small SUVís have made the public aware that small wheelbase vehicles with high centers of gravity are more susceptible to roll-overs than vehicles with large wheelbases and low centers of gravity. Figure 3-9 shows how this may occur on an incline, but how does it happen on a flat surface when turning?

In his *Principia*, Sir Isaac Newton was among the first to document observations of a property common to all matter which we now call Inertia. He stated that bodies at rest like to stay at rest, and bodies in motion like to stay in motion going the same rate in a straight line. To move a resting body or alter a bodyís motion requires the application of an external force. This is summed up nicely in the equation Force = mass x acceleration, or F = ma.

When you are driving down the road and turn the steering wheel, the direction your car is traveling changes. The road pushes against the tires of your car causing it to accelerate sideways. Inertia dictates that your car and everything inside it resist this acceleration with a force equal to their mass times the acceleration. The inertial force is located at the carís CG and is pointing in the direction opposite the acceleration. Because CG is not down on the road where the wheel force is applied, this inertial force creates a torque (or moment) which tries to tip the car over.

Luckily, the weight of the car is generating an opposing moment that works to prevent it from tipping over. All is well as long as the moment due to gravity is greater than the moment due to the inertial force. Wide wheelbase cars with lower centers of gravity are safer because the inertial forces required to roll the car over are much greater. The tires are incapable of generating such large forces, therefore, the car skids instead of rolling over.

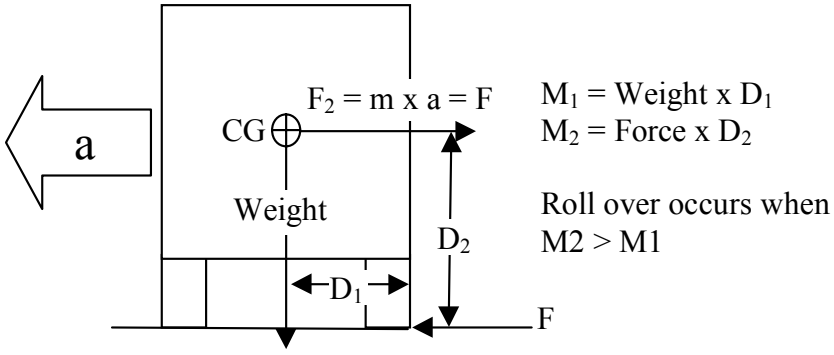


Figure 3-10. Turning Generates Forces and Moments

Inertial forces are also the cause of the ì wheeliesî to which so many FLL robots are prone. When your robot is stopped and you turn on the motors, they initially generate their maximum torque. If the resulting acceleration is large enough, the inertia generated moment will tip the robot backwards, pulling the front wheels off the ground. Luckily the torque output from the motor drops off significantly once the wheels start turning, stopping the wheelie almost as quickly as it started.

Figure 3-11. FIRST Team 254's Robot "Cheesy Poofs" Does a Victory Wheelie

Wheelies should be avoided because it is difficult to control a robot predictably without all of its wheels on the ground. From the turning discussion, we saw that the likelihood of tipping is reduced by having a lower center of gravity or a wider wheelbase. This is also true for wheelies, which are less likely to happen if the center of gravity is low and well ahead of the driven wheels. You can move the CG forward by repositioning components or by adding extra weight to the front of the robot. A similar effect is achieved by moving the driven wheels towards the back.

Question: What else can be done to prevent wheelies besides moving the CG or the wheels?

3.4 Wheel Loading and Friction

The weight of the RCX brick, batteries, three motors, a light sensor, and a rotation sensor is close to 1 pound. Add to this a frame, motor supports, gears, and some sort of manipulator, and a typical FLL robot (if there is such a thing) is supporting 1.5 to 2 pounds on its wheels. Depending on the relative placement of the wheels and supporting frame components, the force on the axles may be many times greater. These high forces can cause much of the motor power to be wasted trying to overcome the resulting friction. What is generating these forces, and how can they be minimized?

Figure 3-12. Cantilevered and Fully Supported Wheels

Most wheeled vehicles use some sort of cantilevered assembly to attach the wheels to the vehicle. Supported on only one side, the axle acts like a lever, generating a moment that has to be reacted to by the frame. A moment is the product of a force (the weight being carried by the wheel) and its perpendicular distance from its axis (the place where the axle is supported by the frame). The farther the wheel is from the frame, the larger the moment.

The way that the frame reacts the moment has a great effect on the amount of friction. Where the axle passes through the frame it does not rest against the top of the channel, but instead it is tilted, contacting the frame at only two points. The farther apart the two contact points are, the lower the forces exerted against each.

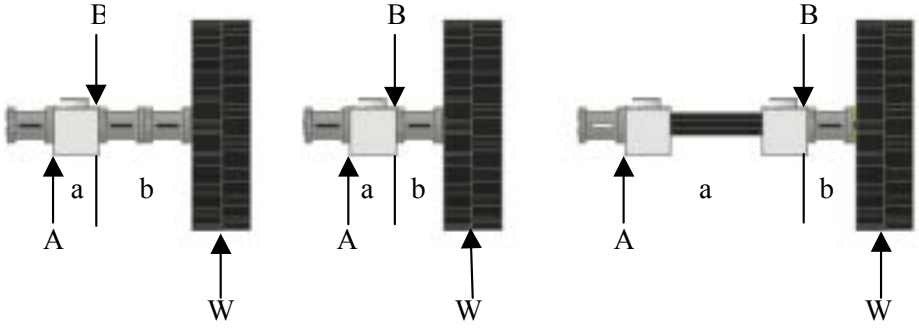


Figure 3-13. Wheel Loading

Distance a	Distance b	Force A	Force B	Friction Forces
1 stud	2.5 studs	2.5 W	-3.5 W	6 W
1 stud	1.5 studs	1.5 W	-2.5 W	4 W
4 studs	1.5 studs	0.375 W	-1.375 W	1.75 W

Figure 3-13 and the corresponding table demonstrate the relationship between wheel placement and loading. ìWî is the weight of the model supported by the wheel. ìAî and ìBî are forces exerted against the axle by the frame. ìaî is the distance between the two contact points on the frame. ìbî is the distance from the frame to the wheel.

To calculate the values of forces A and B we need to use what we have learned about forces and moments. From Newtonís equation, F = ma, we know that a body will accelerate if acted upon by a force. Since our wheel is not accelerating up or down, we can deduce that the forces A, B, and W must all cancel out; their sum is equal to zero. For our purposes here, we will say forces pushing up are positive and forces pushing down are negative.

A - B + W = 0
B = A + W

A moment is just the product of a force acting at a distance, and it follows the same rules as regular forces. A body acted on by a moment will experience an angular acceleration. If the angular acceleration of a body is zero, the sum of the moments about any point on that body must also be zero. To make the equations easier to work with, I am going to pick the point where force B is applied. Here, we will use the convention that a moment that wants to cause a clockwise rotation is positive, and a counter clockwise rotation is negative.

A x a + B x 0 ñ W x b = 0
A x a = W x b
A = b/a x W

Now we can stuff in some numbers. The values below are from the left most example in Figure 3-13.

a = 1 stud
b = 2.5 studs
A = b/a x W
= 2.5 x W
B = A + W
= 2.5 x W + W
= 3.5 x W

The important thing to take away from this discussion is that you can reduce friction by:

1. Placing the wheels close to the frame
2. Spreading out the axle supports to reduce the forces required to react the moment from the wheel.

Question: What would the table entry look like for the fully supported wheel assembly shown in Figure 3-12?

4 Lego Electronics

4.1 RCX Brick

The RCX is your robotís brain, a tiny computer shaped like a LEGO brick. At the core of the RCX is a Hitachi H8 8 bit microcontroller running at 5 to 20 MHz with 32K of RAM. This may seem anemic when compared to modern desktop computers with 2GHz 32 bit processors and 512 Mbytes of memory, but itís as powerful as the Apple II computer I learned to program on and significantly more powerful than the computers that sent men to the moon.

The microcontroller is used to control three voltage outputs, three sensor inputs, and an infrared serial communications port. Snap-on wire leads are used to connect the RCX to motors, lamps, touch sensors, light sensors, rotation sensors, etcÖ The serial communication port is used to download programs from a PC and can be used to communicate between two RCXís.

The RCX is powered by 6 AA batteries. Older versions of the RCX also had a 9V DC input plug for use with an optional AC adapter. If you have this model of RCX, the transformer will pay for itself in saved battery costs. Having two or three sets of good quality rechargeable batteries is also a good idea. The rechargeable alkaline batteries seem to work best with the RCX.

Construction Note: When building your robot, remember that you will have to occasionally remove the RCX to replace the batteries. Design accordingly.

Figure 4-1. The RCX Programmable Brick

4.1.1 Firmware

The RCX contains 32K of battery-backed RAM. Most of this is used up by the firmware that you have to install prior to loading your own robot programs. An additional 6K is reserved to store up to five user programs. This may not seem like much space, but most

RCX programs are only a few hundred bytes long. The remaining memory is used for running programs.

When a program is written for the RCX, it does not contain native code for the CPU. Instead, it is made up of special bytecodes which are interpreted by the firmware. This is similar to the way Java programs on the web are interpreted by the Java Virtual Machine running in your browser. Using a bytecode interpreter allows the RCX to maintain a reliable environment for executing user programs. Your program may not do what you want, but it wonít crash the RCX.

Because the firmware is stored in RAM, it is erased if your RCX remains without power for more than a few seconds. A capacitor in the RCX powers the memory long enough for you to change the batteries (15-30 seconds). If after changing batteries your RCX no longer functions properly, you may have lost your firmware. You can tell if the firmware is loaded by looking at the LCD display. If all you can see is the program number and the minifig icon (either standing or walking as in Figure 4-1), then the firmware is not loaded. This will be the condition of your RCX brick when you receive your kit.

Construction Note: Whenever it is turned on, the RCX is receptive to messages coming in on its infrared serial port. With all the activity and congestion at competitions, it is quite possible that another team could unintentionally overwrite the programs stored in your RCX. Care should be taken to block the IR port when not in use.

4.1.2 Programming

The RCX is programmed using special software that resides on your PC. The program is translated into bytecodes and transferred to the RCX via an infrared serial link (using the IR tower). This method of programming--writing code on one type of computer to be run on another type of computer--is called cross development. A computer without a keyboard and monitor, like the RCX, is called an embedded system. That means all FLLíers are embedded systems developers, familiar with cross development tools. This is impressive stuff to put on your resume.

User programs can be written either in RCX Code or ROBOLAB. RCX Code is the graphical programming tool that LEGO supplies to program the RCX. It is targeted to people with no programming experience. Programs are written by picking instruction blocks and snapping them together, much like using actual LEGO bricks. The original versions of RCX Code were very limited (V1.0 and V1.5). Many instructions supported by the bytecode interpreter were not available, and support for variables was minimal. If you plan to use RCX Code, make sure you have the version 2.0 software. RCX Code is currently only available for Windows PCís.

Figure 4-2. RCX Code Screenshot

ROBOLAB is a product of LEGO DACTA, the education branch of LEGO. It is based on LabVIEW, a graphical programming environment that is really used by engineers and scientists for measurement and control. In fact, NASA scientists used LabVIEW to monitor the Sojourner Rover on Mars! Like RCX Code, ROBOLAB uses a graphical metaphor for writing programs. Command icons are selected from a palette and placed in the programming workspace. Then the command icons are wired together to specify the program flow.

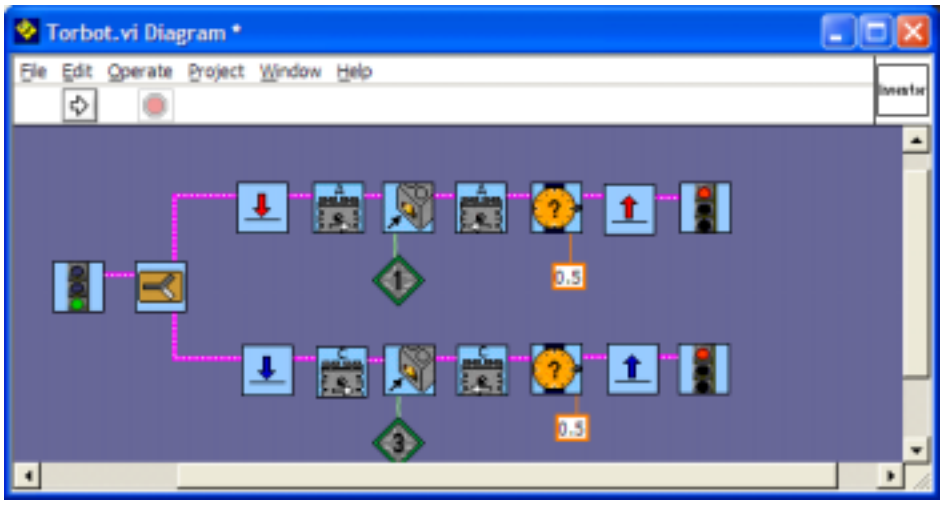


Figure 4-3. ROBOLAB Screenshot

ROBOLAB is a more powerful tool than RCX Code, but not all this power is directed towards programming robots. Large portions of ROBOLAB are devoted to using the RCX like a lab instrument. You can create programs that run on the PC but collect sensor data from the RCX. Other ROBOLAB tools are available to process and analyze the data. There are even tools for publishing the results or converting them to a web

page. Learning all there is to ROBOLAB could take a long time. Luckily, everything you need for FLL is isolated in one section called Inventor. ROBOLAB also has extensive user documentation built into the tool, plus there is a very thorough user manual available online at http://www.ceeo.tufts.edu/robolabatceeo/documentation.htm .

Programming using RCX Code or ROBOLAB is pretty much the same. The programmer uses the mouse to select from a list of icons that represent program instructions. The selected instruction icons are placed into a workspace. Some sort of graphical connection metaphor is used to define the flow of the program (snapping together for RCX Code, wires for ROBOLAB). When the programmer finishes laying out the instructions, he/she uses a menu or toolbar button to compile the program and download it to the RCX.

If you are programming on a Mac, then you have to use ROBOLAB. Otherwise, the choice is mostly a matter of taste. Careful analysis of Minnesota FLL competition results over the last three seasons shows no correlation between a teamís success and the programming language it used.

4.2 Motors

The motor that comes with the challenge kit is called the 9 volt geared motor. It has an internal gear train that delivers a 12:1 gear reduction. This converts an internal rotation speed of 4200 rpm to a more useable 350 rpm[5] at the output shaft. The internal gearing also increases the stall torque to 8.9 Ncm or 0.06 ft-lbs[6].

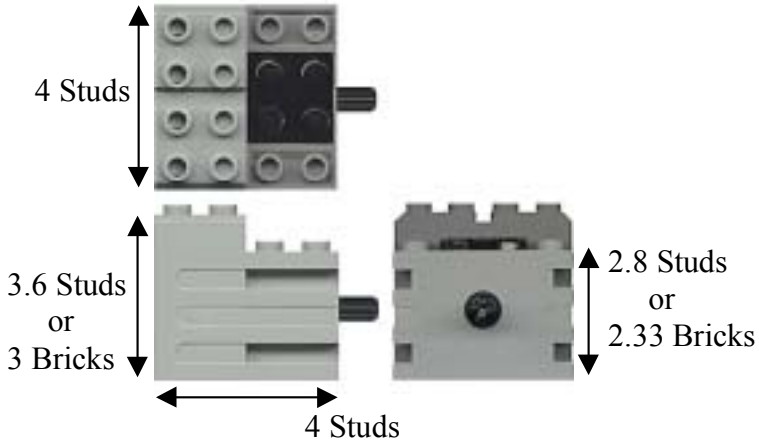


Figure 4-4. 9 Volt Geared Motor

The motor normally draws about 10 milliamps (mA) of current under no load. This can easily increase to 200 mA or more when it is working hard. Add powered sensors (light and rotation), the LCD display, the processor, and a typical FLL robot can be drawing 600 ñ 700 milliamps. Since good quality AA batteries are rated around 2500 milliamp hours (mAh) you can expect to change batteries after three to four hours of operation.

[5] Typical speed under load is closer to 200 rpm
[6] From the Random Hall Lego Robotics Seminar (web.min.edu/sp.742/www/motor.html)

4.2.1 Modes

The RCX has three modes of controlling the motor--Off, On, and Float. In Float mode, the motor is not driven and can spin freely. It behaves just as it would if the connecting leads were removed.

In Off mode, the RCX internally shorts the contacts of the output connector. This has a braking effect on the attached motor causing it to quickly stop spinning.

In On mode, the RCX sends the motor either +9V or -9V to make the motor spin clockwise or counter clockwise. Pulse width modulation (PWM) is used to provide eight power level settings.

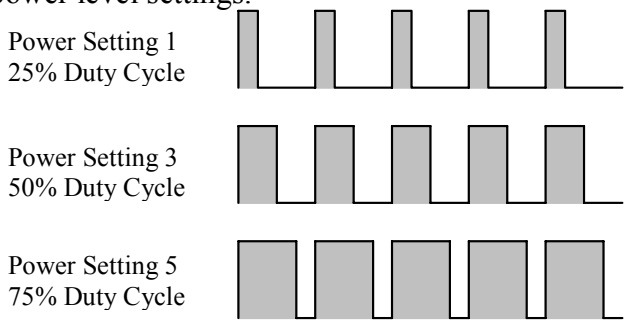


Figure 4-5. PWM Duty Cycles

Pulse width modulation is an inexpensive way to control power output. Instead of setting the output voltage (voltage regulators are expensive), the RXC quickly switches the power on and off. Different power levels are achieved by varying the percentage of time that the power is on (this is called the duty cycle). On the RCX, the minimum power setting is zero, which corresponds to a duty cycle of 12.5% (the power is on 12.5% of the time). At level 7, power is supplied continuously (100% duty cycle).

The geared motor has a little internal flywheel that evens out the motor speed as the power is switched on and off for PWM. You see it in operation when you spin the shaft of an unattached motor. The shaft completes several revolutions while the rotation speed slows down to zero. If you continue to give the shaft a twist once a second or so, it is pretty easy to get it turning at a fairly consistent speed. This is kind of a mechanical version of pulse width modulation

Once you get the motor spinning, it doesnít take much energy to keep it going. This can be a bit of a problem if you are using low power settings to run your motor slower. Without much load, a geared motor spins up to about the same speed regardless of the power setting. Each power-on pulse supplies more energy than friction can dissipate during the rest of the cycle. The extra power gets stored in the flywheel as inertia. With every cycle, a little bit more energy is stored in the flywheel, increasing the rotation speed until equilibrium is reached at the point where the motor torque starts to fall off.

Figure 4-6. Using a Pulley to Increase Drag

Motor speed control is reestablished by adding some friction. After all the careful thought and attention to detail you put into your robot design to minimize friction, now itís time to have some fun. Take out your list of good building practices and break all the rules. Squeeze the bushings up tight against the sides of the beams. Use three or four gears instead of two. Be creative, but make sure the judges know the reason and thinking behind your modifications. You donít want to be penalized for poor mechanical design.

4.2.2 Attaching

A motor is attached to one of the three output connectors by using a snap-on wire lead. You can reverse the rotation direction of the motor by changing the orientation of the connection. If you need to wire up a motor during competition for a configuration change, make sure that you always attach the leads the same way. Marking the connector leads and motor ports with color coded 1 x 2 plates works well as a reminder.

Figure 4-7. Using Color Coding to Document Proper Connector Orientation

Motors generate significant force and will work themselves loose if only held in place using snap-on mountings. Additional support is required. This can be provided using the same cross bracing techniques used to build strong structures.

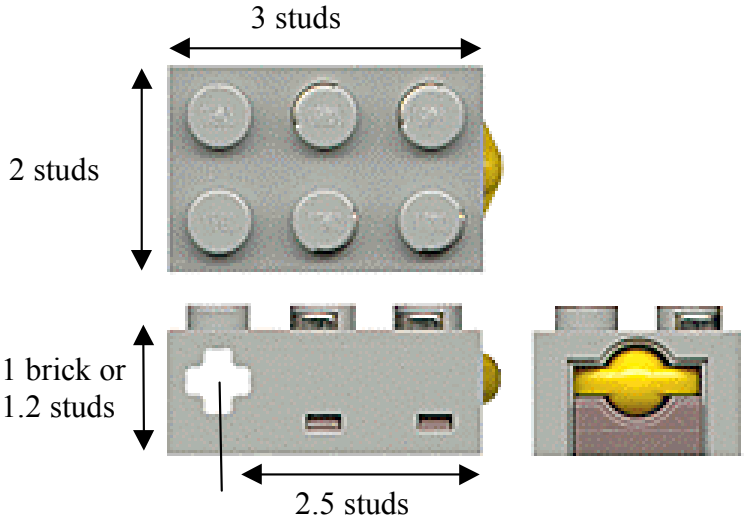
Figure 4-8. Strengthening Motor Mounts with Cross Bracing

The RIS kit includes 1 x 2 plates with rails that can be used as brackets for the motors. They provide a compact and sturdy way to attach a motor to a beam. The rails can also be used to create a quick release motor mount. The motor in Figure 4-9 is easily removed after unsnapping the bottom 2 x 6 plate that holds it in place. This could be useful if your robotís design requires you to move a motor during head-to-head competition.

Figure 4-9. Motor Mount Using Rails

4.3 Touch Sensor

The touch sensor is the simplest to use of all the LEGO sensors. It consists of a 2 x 3 brick with a spring loaded button. The button varies the resistance of the sensor when it is pressed. The RCX software converts the sensor feedback to a binary signal: one for when the button is pressed, and zero for when it is released.



The touch sensor is unusual in that it doesnít have an attached wire lead like the rotation and light sensors. Instead, it uses a snap-on lead like the motors. The four studs closest to the button have electrical contacts to receive the lead wire connector. Orientation is not important if the lead wire is attached to all four contact studs. You can also make the connection by using only two of the contacts, but when doing so, the lead wire must be oriented as shown below.

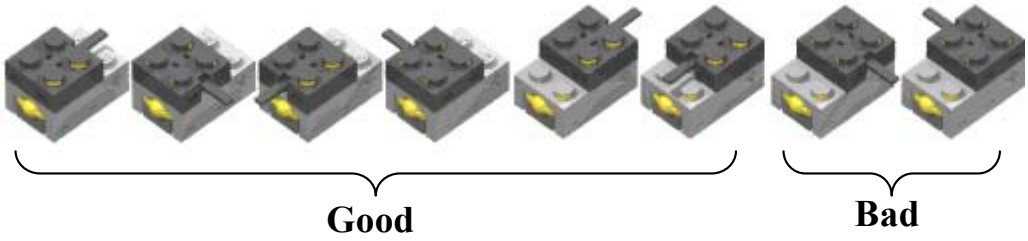


Figure 4-10. Wiring the Touch Sensor

4.3.1 Bumpers

A bumper is a device that notifies your robot that you ran into an obstacle. When struck, the bumper moves and presses or releases a touch sensor, notifying your robot of the impact. Bumpers are the most common use for a touch sensor in FLL. This may be due to the Constructopedia using touch sensors primarily in this role, or it could be because bumpers are such useful devices.

Figure 4-11. A Simple Bumper

A bumper assembly usually consists of four parts: the bumper, a sensor, a return mechanism, and a supporting framework. The bumper component is the part that receives the impact and converts it to motion that can be detected by the sensor. The motion is usually sliding (Figure 4-11) or pivoting (Figure 4-12), though I have seen some interesting bumpers built of flexible parts that use bending. The sensor component is usually a touch sensor, but bumpers can be built using rotation sensors or light sensors.

Figure 4-12. A Bumper that Uses the Rotation Sensor

The return mechanism forces the bumper to its normal (non-impacted) state when the obstacle is removed. The most common return mechanism is a rubber band that pulls the bumper against a mechanical stop. The design of some bumpers allows gravity to be used as the return mechanism. The simple bumper in Figure 4-11 originally used the spring inside the touch sensor as the return mechanism. That design proved to be unreliable, occasionally signaling a collision when none had occurred. Adding the rubber band fixed the problem.

Figure 4-13. A Normally Closed Bumper Design

Most bumper designs can be divided into two categories based on the way they monitor their sensor for impacts. The bumper in Figure 4-11 is a ìNormally Openî design. When a collision occurs, the bumper presses against the touch sensor, closing the contacts. In a ìNormally Closedî style of bumper, the return mechanism pulls the bumper against the touch sensor. A collision causes the bumper to pull away from the touch sensor, opening the contacts.

When the normally open bumper in Figure 4-11 collides with an obstacle, the impact of the collision is transmitted through the bumper to the touch sensor. This will loosen the sensor mounting, and after a few collisions the sensor may fall off. Additional bracing could be used to strengthen the sensor mounting, but this just transmits the impact to the robot frame.

In a normally closed bumper like that in Figure 4-13, the force of a collision is absorbed by the rubber bands. This bumper also has a long ìthrow,î the distance between where a collision is detected and the bumper ìbottoms outî against a hard stop. The long throw allows the bumper to sense an obstacle and stop the robot before a hard collision occurs.

Figure 4-14. Improved Normally Open Bumper

Normally closed bumpers are good at detecting obstacles and protecting your robot from forceful collisions. But the normally closed touch sensor causes problems for sensor stacking (attaching multiple sensors to one input port). A closed touch sensor will force the input to be 1 (if configured as a touch sensor port) or 100 (if configured as a light sensor port). The improved bumper above retains all the benefits of a normally closed design with the advantage of having the touch sensor be normally open.

4.3.2 Limit and Position Switches

Touch sensors have many uses other than bumpers. Limit and position switches tell you when a component of your robot is in a certain location. They are discrete sensors (as opposed to continuous sensors) in that they provide position information only at certain locations. You only know where you are while the touch sensor is pressed. When it is open, you only know where you are not.

Figure 4-15. Position and Limit Switches

Limit and position switches really differ only in their use. Limit switches are guards that watch for something being where it is not supposed to be. They are protective mechanisms used when damage could result from a device moving outside its working envelope. A position switch is just the opposite; it is a messenger that notifies you when a device is where you want it to be.

The light scanner in Figure 4-15 has two limit switches and a position switch, and all use only one touch sensor! The bottom two cams are part of the limit switches. They prevent the scanner post from twisting more than +/-90 degrees. The top cam is a position switch that tells the RCX when the sensor is pointed straight forward. To be able to interpret the touch sensor feedback, the program has to keep careful track of where the scanner post is and which way it is spinning.

4.3.3 Rotation sensor

Perhaps the most repeated gripe Iíve heard from FLL teams is about the lack of a second rotation sensor. Having two just seems to make sense: one for left and one for right, or one for distance and one for steering. Other teams see the lack of a second rotation sensor as a challenge, and some of their solutions have been very creative.

Figure 4-16. A Touch Rotation Sensor

Just as a rotation sensor can be used in place of a touch sensor as part of a bumper, a touch sensor can be used to measure rotation. In the rotation sensor, opto-interrupters use light to monitor the position of a fan-shaped rotor. In Figure 4-17, the touch sensor is used to monitor the position of the axle joiner.

Question: What is the resolution of the rotation sensor in Figure 4-16?

4.4 Light Sensor

The light sensor is a 2 x 4 LEGO brick that contains a red light emitting diode (LED) and a photo transistor. The red LED illuminates the area in front of the brick, and the photo transistor measures the intensity of the reflected light. The light sensor returns a value in the range of 0 to 100%. The lowest reading I have ever seen is 20%, which I took at midnight in the middle of my basement with the lights off. You can get to 100% by aiming the sensor at the sun on a clear day or by holding it within a few inches of a 100W light bulb. In normal operation, the sensor values tend to be between 30 and 60%.

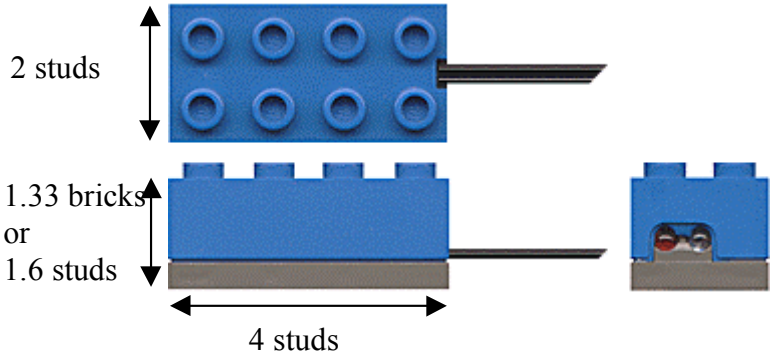


Figure 4-17. The Light Sensor

One of my earliest experiences with the light sensor was when I used it as part of a candy sorter. I wanted to sort bags of peanut M&Ms based on the color of their candy shells (brown is my favorite). The concept was simple. I would use the light to determine the color, then move the robot to position it over the correct bin.

I made a pivoting ì Lazy Susanî type robot that I could surround with cups to receive the candy. The rubber track made a nice conveyor belt, and I found a design for something called a ì marble pumpî that did a great job of getting the candy, one at a time, out of the hopper. With all the mechanics working correctly, all that was left was to pop the light sensor values for each color into the program. Visions of watching Battlebots while munching on brown M&Ms danced through my head.

Table 4-1. Light Sensor Readings for Peanut M&Ms

Distance From Sensor	Red	Orange	Yellow	Green	Blue	Brown
Ω stud	48	48	51	38	40	40
1 stud	45	46	48	37	35	35

I never got that sorter to work very well because the sensor values for different colored M&Ms were too similar. After playing around a bit and trying a few things, I got it to sort M&Ms into three color groups: Yellow in one cup, Orange and Red in another, and Blue, Green, and Brown in a third. Eventually, I got tired of working on the robot and disassembled it for parts. I think this same sort of thing happens to many FLL teams that try to use the light sensor on their robot. They come up with a really neat idea and build robot to try it out. During testing, the robot doesnít work quite right or doesnít work reliably. The team tinkers around with it a bit, fails to make real progress, and tosses the sensor back into the parts bin.

The problems I had with the candy sorter were not the fault of the light sensor. The project was doomed from the very start when I chose to use the sensor in an application for which it was ill suited. I needed to have a better understanding of the light sensor before using it in another project, therefore, to increase my understanding, I performed a series of experiments.

4.4.1 Experiment #1, Color

From my candy sorter experience, I knew the light sensor reading is affected by the color of the object at which it is aimed. But the reading also appears to be affected by the distance between the sensor and the object, variations in the objectís size and shape, and ambient lighting conditions. If I was going to learn anything about the light sensor, I had to limit the number of variables. For my first experiment, I decided to study the relationship between light sensor value, color, and distance.

Figure 4-18. Light Sensor Color Experiment

For this experiment, I built a box out of LEGO parts with a light sensor centered at one end. The box had solid sides and a top and bottom to prevent ambient light from distorting the results. Next, I made several different colored ì targetsî that I could place in the box at various distances from the sensor. I used 1 x 4 bricks to make the targets because I have them in many different colors, and they will allow others to easily reproduce my work. For each different color, I took sensor readings at 1, 2, 3, 4, and 5 studs spacing. The results are shown in the table below.

Table 4-2. Light Sensor Readings for Color Experiment

	Distance (studs)				
	1	2	3	4	5
Black	43	38	35	33	27
White	54	48	43	40	37
Gray	49	43	39	35	32
Dk Gray	45	40	35	31	29
Yellow	53	48	43	39	36
Red	50	45	41	37	35
Blue	44	38	34	32	31
Green	42	37	32	29	28

4.4.1.1 Color

As I was performing my light sensor experiments, it became apparent that the light sensor doesnít see light the same way I do. To my eyes, the green bricks appear brighter than the red bricks, yet the green brick sensor readings were consistently lower. In fact, the light sensor readings for the green, blue, dark gray, and black bricks were essentially the same.

Light is made up of electromagnetic waves--vibrations of electric and magnetic fields that travel through space. There are many types of electromagnetic waves: some we can feel (heat), some we can see (light), and some we have harnessed for our own uses (television and radio). Our eyes can detect electromagnetic waves with wavelengths in the range of 400nm (violet) to 700nm (red) in length. We call this tiny portion of the electromagnetic spectrum visible light. Our eyes perceive different wavelengths of visible light as having

different colors. Light with a wavelength of 700nm is red; blue light has a wavelength around 460nm.

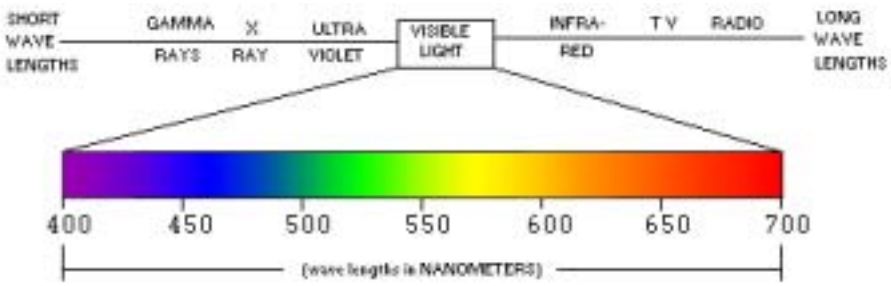


Figure 4-19. Visible Light Spectrum

The LEGO light sensor uses a silicone photo transistor as its sensing element. According to Michael Gasperi[7], the photo transistor is most sensitive to light in the red to infrared range, having peak sensitivity around 800nm. Cheap red LEDs of the type used in the light sensor, output a fairly broad spectra, but the most intense light is somewhere between 650 and 750nm. This makes it a decent match for the photo transistor. Our eyes are most sensitive in the blue-green to yellow range (500nm ñ 600nm). This explained why the green bricks looked so bright to me, and the red bricks looked bright to the light sensor. But it didnít adequately explain why the light sensor readings for the yellow bricks were so high, and the readings for the green bricks were so low.

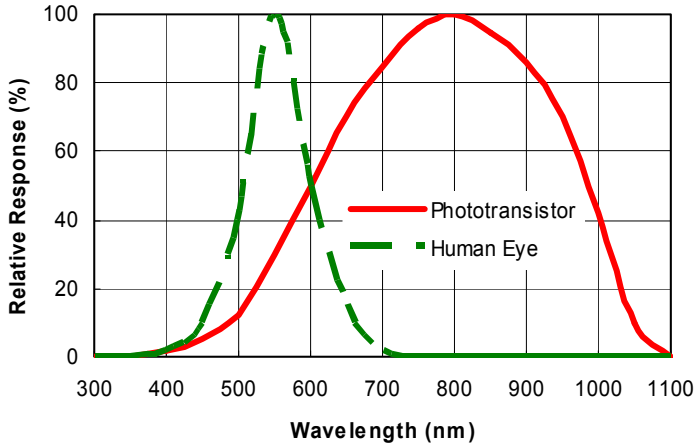


Figure 4-20. The Light Sensor Sees Differently than Our Eyes

There are two ways in which we see colors. One way is for an object to emit light waves in the wavelength of the observed color. The light sensorís red LED emits light with a wavelength around 660nm, which is why it appears red. Your computer monitor and television can emit red, green, and blue colored light. Other colors are created by mixing these primary colors. The other way to see color is for an object to reflect light waves in the frequency of the observed color and absorb all other wavelengths. The red LEGO brick reflects red light, but absorbs blue and green light.

[7] ìExtreme Mindstorms An Advanced Guide to Lego Mindstormsî, Dave Baum, Michael Gasperi, Ralph Hempel and Louis Villa

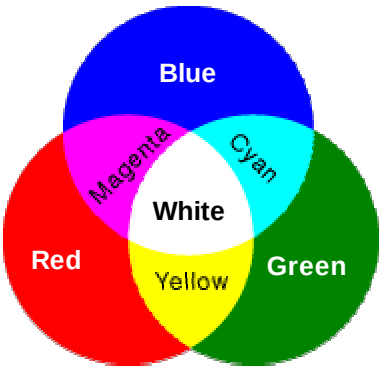


Figure 4-21. Light Colors

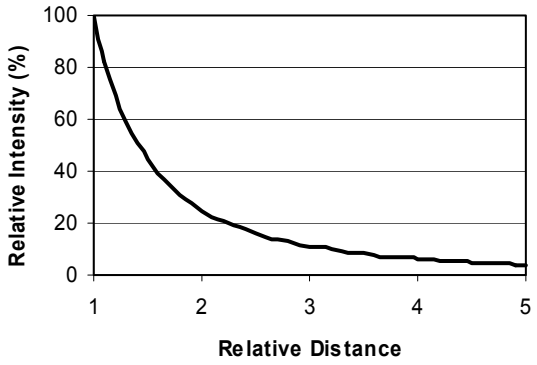
The light sensor readings in Table 4-2 start to make sense when you think of color in terms of emission, reflection, and absorption. With the cover placed on my test box, the only light striking the colored brick target was the red light emitted from the red LED. The red LEGO bricks are very good at reflecting red light, thatís why they appear red to us and bright to the sensor. The green and blue bricks donít reflect red light well at all and have sensor readings that would make you think they were black. The yellow and white bricks give the highest sensor readings because they reflect red light and other wavelengths the LED emits as well.

Question: In a dark room, what color do you see when shining the light sensor's red LED light on a yellow LEGO brick?

4.4.1.2 Distance

The Inverse Square Law is a neat mathematical formula that shows up all over the place in physics. It defines the relationship between the measured strength of a thing, like the pull of gravity or the loudness of sound, and the distance you are from the source of the thing. The Inverse Square Law for light intensity states that the intensity of light observed from a source of constant brightness falls off as a square of the distance from the object. It is generally expressed as the ratio of light intensities l_1 and l_2 at distances d_1 and d_2.

$$\frac{l_1}{l_2} = \frac{d_2^2}{d_1^2}$$



This means that if we double the distance to a light source, the observed intensity is decreased to $(1/2)^2$ or ° of its original value. At 3, 4, and 5 times the distance, the

intensity would be decreased to $^1/_9$, $^1/_{16,}$ and $^1/_{25}$ of the original value. Initially, intensity falls off very quickly as you move away from a light source. But as you get farther and farther away, the change becomes less and less noticeable.

Given that, it was no surprise that plots of the sensor readings show the intensity decreasing rapidly as the distance between the light sensor and the target increases. Iím not sure why the sensor readings didnít fall off as quickly as predicted by the Inverse Square Law, but I think it might be due to the photo transistor and LED being so close together (much of the light measured by the photo transistor never leaves the light sensor housing). Looking at the plots and the table, I now see why I never got my candy sorter to work very well. A change in distance as small as half a stud is enough for the light sensor to confuse red with yellow or blue with green, and if a red brick is placed two studs farther away than a green brick, the light sensor canít tell the difference between them.

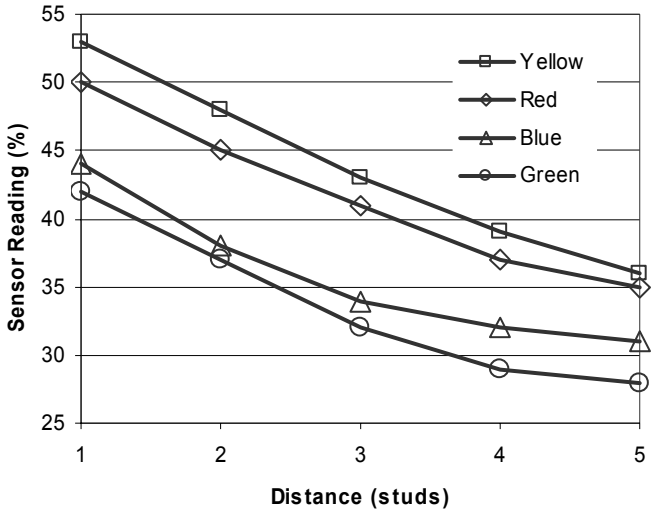


Figure 4-22. Color Experiment Sensor Readings

4.4.1.3 Conclusions

From this experiment, I drew a few conclusions about the LEGO light sensor. The most important is that you can use the sensor to measure distance if you know the color of the reflecting surface, or you can tell color if you know the distance from the reflecting surface, but you canít do both. For best results, the light sensor should be used in carefully controlled conditions. Care must be taken to limit the number of variables.

The light sensor does not see light the same as humans do. The sensitivity of the sensor is not well matched to human vision. What appears bright to our eyes may be dark to the light sensor, and light that is outside our range of vision can easily drive the sensor reading to 100%. It is always a good practice to collect and analyze some sensor data prior to developing a strategy for using the light sensor.

I also learned that the light sensor is extremely sensitive to the distance between the sensor and the reflecting surface. When using the sensor to measure contrast it must be held a fixed distance from the reflecting surface. Even small variations in distance can make the sensor readings unusable. Remember this when designing the light sensor mount for your robot.

4.4.2 Experiment #2, Ambient Light

In my first experiment, I studied the relationship between color and the light sensor reading. From the data, I deduced that the sensor reading is not only dependant upon the color of the object being viewed but also upon the wavelength of the light illuminating the object. In normal usage, the light sensor and target will not be in a sealed container. Light from external sources, ambient light, will have some effect on the sensor reading. To better understand how ambient light affects the sensor readings, I performed a second experiment.

For this experiment, I built a wall of different colored bricks to use as a target and an adjustable stand to hold the sensor a known distance away from the bricks. The stand was ì spider likeî to minimize the shadow it cast. I then recorded the light sensor readings for each colored brick at different distances under a variety of lighting conditions. First, as a baseline, I took measurements in a completely dark room in my basement (light sensor reading 22%). I repeated the experiment in ì normal basementî lighting conditions (sensor reading 46%) and then in ì bright competitionî lighting conditions (sensor reading 65%). The readings appear in Table 4-3.

Figure 4-23. Ambient Light Experiment

The first thing I noticed from the data was how the green and blue sensor readings were the most affected by the presence of ambient light. The incandescent lamps used to produce the moderate and bright lighting conditions emit a full spectrum of visible light wavelengths. This gives the blue and green bricks something to reflect, so they appear much brighter to the sensor than in the previous experiment.

Question: In the '01 FLL challenge "Volcano Panic," a blue tape line was used to mark the edge of the ocean. The intersection of the blue

line with one of the many black lines on the table was a useful navigation landmark. Many teams had a hard time finding the blue line. What was the likely source of their problem, and how could they have improved their chances?

Table 4-3. Sensor Readings for Ambient Light Experiment

	No Light				Low Light				Bright Light			
	1 stud	2 std	3 std	4 std	1 stud	2 std	3 std	4 std	1 stud	2 std	3 std	4 std
Black	39	27	27	26	39	36	35	35	39	46	52	52
Dk Gray	40	34	30	28	40	38	38	38	43	47	53	53
Gray	43	38	34	33	45	45	43	43	49	53	57	57
White	50	44	40	37	52	50	50	50	58	60	62	63
Green	36	32	28	27	40	42	42	42	49	55	58	58
Blue	39	36	33	29	45	44	45	45	51	58	60	60
Red	47	43	39	36	51	49	48	49	56	60	62	62
Yellow	49	44	39	37	52	50	49	50	58	61	63	62

However, the really interesting and important thing to notice is how ambient light changes the relationship between sensor value and distance. Figure 4-24 is a plot of the Gray brick sensor readings under the three lighting conditions. In ì no lightî conditions, the sensor readings are similar to those taken in the sealed chamber in experiment #1. The sensor readings are almost flat in ì low lightî conditions. The light from the red LED is completely overwhelmed by the light from multiple 100W incandescent lamps used to light the room. And in ì bright lightî conditions, the sensor readings actually increase as the sensor is moved away from the target. At shorter distances, the light sensor body and test fixture blocked some of the ambient light from striking the target. Less light was blocked as the sensor was moved farther away.

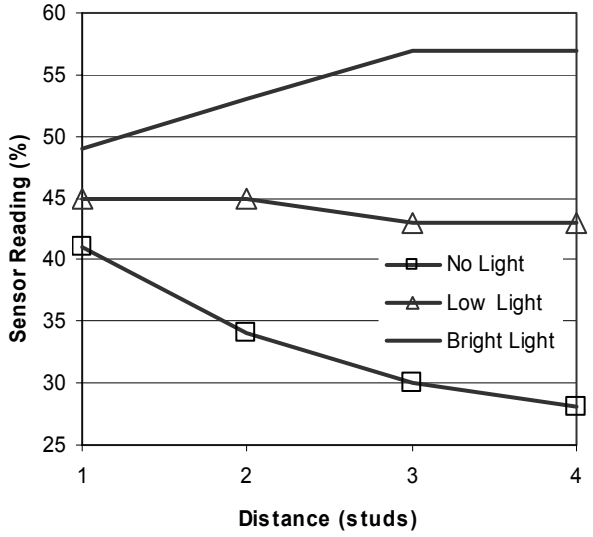


Figure 4-24. Light Sensor Readings for Gray LEGO Brick

4.4.2.1 Conclusions

It appears that the light sensor is very sensitive to ambient light, which is disconcerting because you seldom have control over the lighting. This point was really driven home when I demonstrated a few of my robots at an outdoor exposition. Even though the demonstration table was under a canopy tent, I had a very difficult time getting the robots with light sensors to work correctly. And I thought I had the sensors pretty well protected. I should have tested my robots under the expected lighting conditions. Remember this when testing your robot in a dimly lit basement. It may behave very differently once it is under the bright lights of the competition arena.

4.5 Rotation Sensor

The rotation sensor is used to measure how far a rotating axle has turned. It has a free spinning bushing to receive the axle. As the bushing turns, a counter in the RXC is incremented or decremented. Each full rotation registers as 16 counts giving the sensor a resolution of 22.5 degrees.

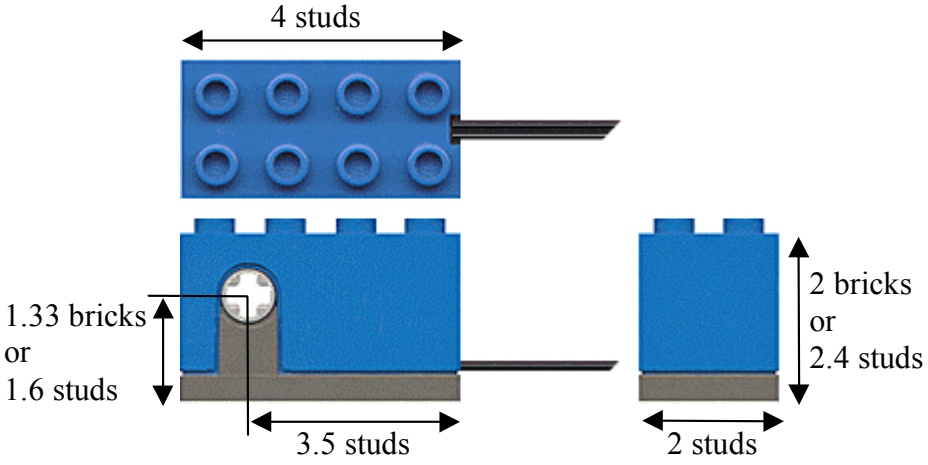


Figure 4-25. Rotation Sensor

The rotation sensors can track up to 32,767 counts in either direction. If you exceed this, the sensor will ì wrap aroundî going from largest positive value to largest negative value (or vice versa). In practice, the count limitation is seldom a serious problem. A robot with the rotation sensor directly attached to the large 81mm balloon tires will travel 1/3 of a mile before wrap around occurs.

4.5.1 Resolution

The resolution of the rotation sensor is usually adequate for distance measurement. Using the largest LEGO wheel (the 81.6 mm balloon tire), 22.5 degrees works out to only 0.6î of linear travel. For the small balloon tire, it is ° î. But this much measurement uncertainty may be unacceptable for accurate turning. For a differential drive robot with a wheelbase 10 studs wide, ° î of wheel travel works out to 8 degrees of turn angle.

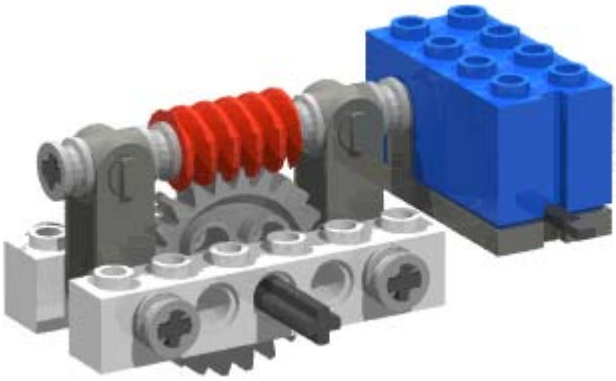


Figure 4-26. Using Gear Reduction to Increase Resolution

Effective sensor resolution can be increased using gear reduction. For the example in Figure 4-26, the input shaft (one with the worm gear) turns 24 times for each revolution of the output shaft (one with the spur gear). The rotation sensor will generate 384 counts for each revolution of the 24 tooth gear. Thatís less that 1 degree per count. Unfortunately, this assembly has about 2 degrees of backlash, causing a lot of that resolution to be wasted.

4.5.2 Internals

The rotation sensor is an interesting little device. If you had Supermanís X-ray vision, you could see that it is built using two tiny opto-interrupters. These are fork shaped devices that have an LED on one tine and a photo resistor on the other. The photo resistor senses when an obstruction passes between it and the LED.

Figure 4-27. Rotation Sensor Internals. DON'T DO THIS!!!

The rotating bushing has four little fan blades that pass between the opto-interrupters as the bushing spins. The opto-interrupter output changes as a blade enters and again as it leaves. This gives 8 transitions per rotation for each sensor. Using two sensors provides 16 counts per revolution plus allows the sensor to calculate the direction of rotation..

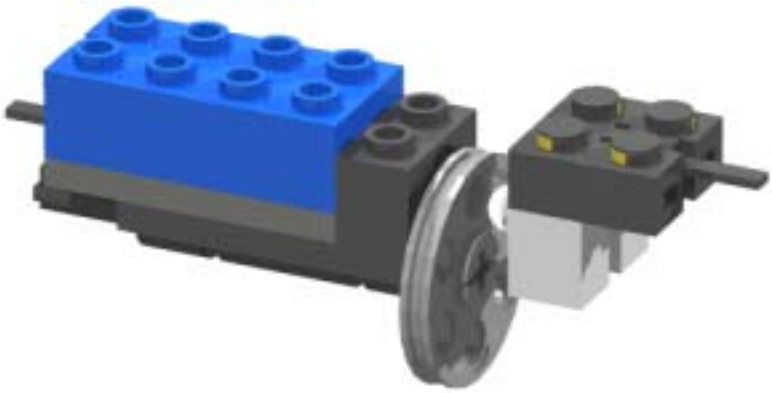


Figure 4-28. Homemade Rotation Sensor Made From LEGO Parts

You can use a light sensor, lamp, and medium pulley wheel to build your own rotation sensor. The light sensor and lamp are used to implement the opto-interrupter. As the axle turns, the pulley wheel will alternately block or transmit light from the lamp. With only one sensor, you cannot determine rotation direction, but direction can usually be ascertained by looking at the motor commands.

Question: What is the resolution of the homemade rotation sensor in Figure 4-28?

4.5.3 Counting Errors

There have been dozens of posts on LUGNET (LEGO Users Group Network) discussing a mysterious problem with the LEGO rotation sensor. There is a wide held belief that counting errors occur if the rotation speed is too fast or too slow. Exact speeds are uncertain, but they have been reported to be below 10-20 rpm or above 1200-1400 rpm. The large body of experimental data available on the web seems to agree that the rotation sensor is quite reliable if speeds are kept in the 60 ñ 1000 rpm range.

Figure 4-29. A Homing Switch to Reset the Rotation Sensor

The rotation sensor can be tricky to use, so donít immediately blame counting problems if you have difficulties using it in your robot. Repeatability problems are more likely due to programming errors (forgetting to clear the sensor before use or interpreting the sensor

readings incorrectly), design errors (inadequate sensor resolution or excessive backlash), control problems (accelerating and turning too fast), or variations in initial conditions (not putting everything in the right place before pushing the start button). But if you still have problems, after checking that everything else works fine, you might need a homing switch.

A homing switch is a secondary sensor that can be used to check or reset the value of the primary sensor. The light scanner in Figure 4-29 uses a touch sensor for a homing switch. The cam attached to the rotating light sensor presses the switch when the light sensor is facing forward. Pressing the switch notifies the RCX to check the value of the rotation sensor and reset the counts if they are incorrect (the count should be zero at the straight ahead position).

4.6 Sensor Stacking

The RCX has three motor output ports and three sensor input ports, yet the challenge kit includes two touch sensors, two light sensors, and one rotation sensor. What do you do with all those extra sensors? Many FLL teams resolve this dilemma by leaving the sensors out of their design. One team came up with the creative idea of using its touch sensors as decorative eyes. The eyes looked cool but werenít very helpful. The most useful solution is to stack the sensors.

Stacking is the attaching of multiple sensors to a single input port. When done correctly, the RCX is able to read the input port value and determine which sensor the value came from. Touch sensors are the best candidates for stacking. They can be stacked with light sensors or other touch sensors. Theoretically, two light sensors could be stacked, but I wouldnít know how to interpret the sensor reading. Rotation sensors cannot be stacked due to the nature of their feedback signals.

Table 4-4. Sensor Readings for Two Stacked Sensors

	Touch Open	Touch Closed	Light Sensor
Touch Open	0	1	light level
Touch Closed	1	1	100
Light Sensor	light level	100	???

From Table 4-4, we can see the importance of planning for sensor stacking in the early stages of robot design. When a touch sensor is pressed, no other sensors attached to that input can be read. Any bumpers or position or limit switches that use a touch sensor must have the sensor be normally open (zero).

5 Robot Drives

A vast majority of robots, especially those used in industry, are of the stationary type. They may have hands to pick things up and arms to move things from here to there, but they have no legs. Instead, they depend on other specialized machines to bring them the materials they need for their work and cart off the fruits of their labor. The robots designed for FLL are quite the opposite; they need to move. They have places to go and scientists to save, and all without any outside intervention or control. This kind of robot is commonly referred to as an Autonomous Robotic Vehicle or ARV, and it is among the most difficult kinds of robots to build and program.

The part of an ARV that is responsible for moving around is called the motion base or motion platform. Early motion bases were exclusively wheeled platforms that only had to move their robots around on a smooth laboratory floor. As robots moved out of the lab, automotive and tank type robots became increasingly popular. Nowadays, ARVís are zooming down the highway at 60mph, diving to the deepest depths of the ocean, walking down into the cauldrons of active volcanoes, flying surveillance over enemy armies, and rolling over the surface of Mars.

The motion bases for FLL robots havenít been quite so exotic. Their operating environment is closer to the smooth laboratory floor than the cauldron of an active volcano. Wheeled platforms are the most common, but tracks are popular too. Some have all-wheel-drive to help them climb inclines or traverse rugged terrain. A few teams have tried automotive type platforms with steered front wheels, but most robots use some sort of differential steering arrangement where turning is performed by changing the relative speeds of the two drive motors.

Much of your robotís success will be determined on how well the strengths of its motion base match the needs of your overall strategy. Whatís more important, speed or accuracy? Is traveling in a perfectly straight line really required? Do your plans require lots of turns and maneuvering in tight quarters? Is pulling capacity or load carrying capacity important? What about obstacles, uneven terrain, surface variations? Many questions must be answered before building can begin.

5.1 Differential Drive

Differential drive motion platforms were very popular in early mobile robots. Two wheels mounted on a single axis are independently powered and controlled, providing both drive and steering. Steering is performed by changing the relative velocities of the two drive wheels. Spinning the left wheel faster causes the robot to turn right. Spinning the right wheel faster causes the robot to turn left. By spinning the wheels in opposite directions, it is possible to turn in place, something that is impossible to do in an automobile. Perhaps the most common example of a differential drive platform in everyday life is the wheelchair.

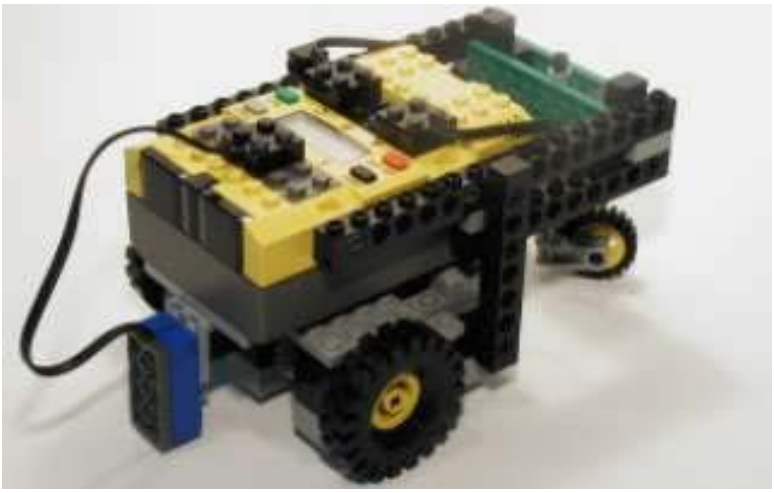


Figure 5-1. A Differential Drive Robot

Differential drive robots are very popular in FLL. Iím sure this is in large part due to their inclusion in the Constructopedia. However, they really are a good platform for robot competitions. Differential drives are easy to design and build, making them a good choice for teams with young children. They are very agile and can be quite compact, allowing them to maneuver in tight spaces. Having two motors attached to the driven wheels makes differential drive robots very powerful and capable of climbing inclines that would stop other robots.

5.1.1 Casters

Because it is hard to make a stable two wheeled vehicle, most differential drive robots have one or more non-driven wheels to provide balance. Commonly called casters, idlers, or bogeys, the job of these wheels is to prevent the robot from tipping over while at the same time having little or no impact on locomotion or steering. Unlike the driven wheels that have only one axis of motion, these balance wheels must be able to turn in any direction. Upside-down ball rollers are perfectly suited for this task. Using skids can also be a good solution if the floor is smooth and slippery.

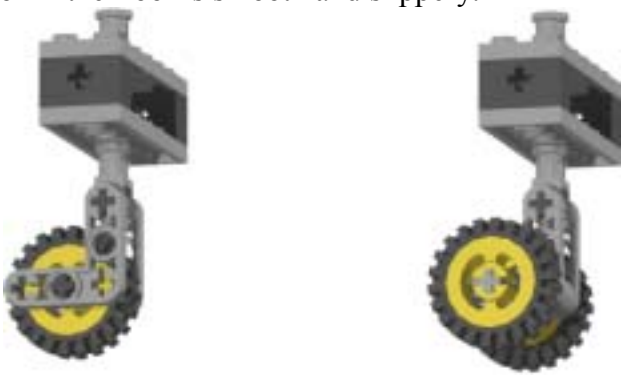


Figure 5-2. Swivel Casters

But LEGO does not make ball bearings, and skids are not always appropriate. Therefore, FLL differential drive robots usually use a swivel caster. A swivel caster has one or more wheels which are attached to a pivoting horn. The horn is bent to offset the wheel axle

and pivot axle. This offset makes the caster wheel self aligning. When your robot changes direction the caster automatically turns to the new heading.

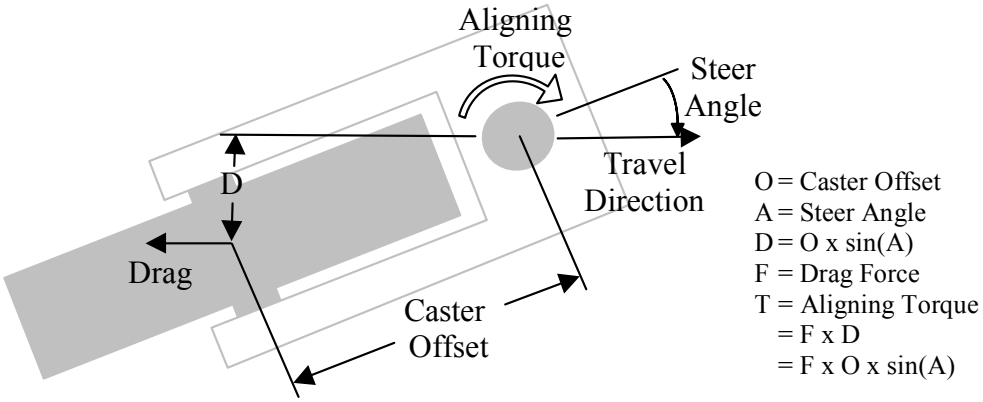


Figure 5-3. Swivel Casters are Self Aligning

As a differential drive robot moves forward, the ground slides backwards beneath the caster wheels. Friction between the wheel and ground generates a force pulling backwards on the wheel. If the caster is pointed in the direction of travel, the drag force just causes the casterís wheel to turn. However, if the caster is not pointing in the direction of travel, it generates an aligning torque which tends to pivot the caster. The amount of aligning torque is proportional to the caster offset. Casters with a large offset will have higher aligning torques for a given steer angle. They are more stable and follow better than casters with small offsets.

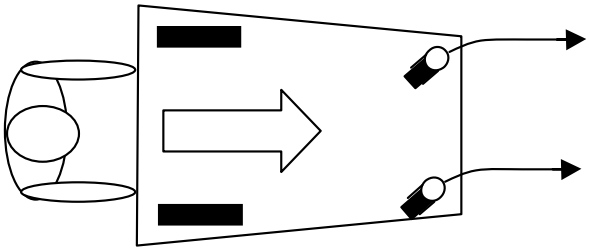


Figure 5-4. Casters Generate Steering Forces While Aligning

Increasing the offset will make a caster less twitchy, but it can also amplify caster steer, which is something that is not very desirable in robot applications. Caster steer is something we are all familiar with from our experience with shopping carts. It is the tendency of the casters to steer the front end of the cart while they align themselves after completing a turn. This is particularly noticeable if the cart is heavily loaded and straightening after a sharp corner or reversing direction. While only mildly annoying in a shopping cart, it can be a real problem in a robot. Excessive caster steer makes your robot difficult to control and introduces errors in distance and steering measurements.

When the caster wheel is not aligned with the direction of travel it generates a sideways force as it rolls, just as the front wheels of an automobile generate a sideways force during steering. If the friction between the wheel and the ground is too great to allow the caster to skid, the force is exerted against the cart, causing the front end to move sideways. As the caster straightens out, the sideways motion becomes less and less noticeable.

Question: Drawing on your shopping cart experience, how can you minimize the caster steer effect?

5.1.2 Wheel Configuration

Early differential drive robots often had a diamond shaped wheel configuration like that shown in Figure 5-5. These robots used encoders (rotation sensors) on the two drive wheels to measure distance traveled and to control steering. Therefore, it was very important that the drive wheels be in firm contact with the ground. By having casters both fore and aft, it was possible to put the robotís center of gravity directly over the drive wheels, which supported almost all of the robotís weight. This maximized the traction of the drive wheels and minimized the forces generated by the casters aligning.

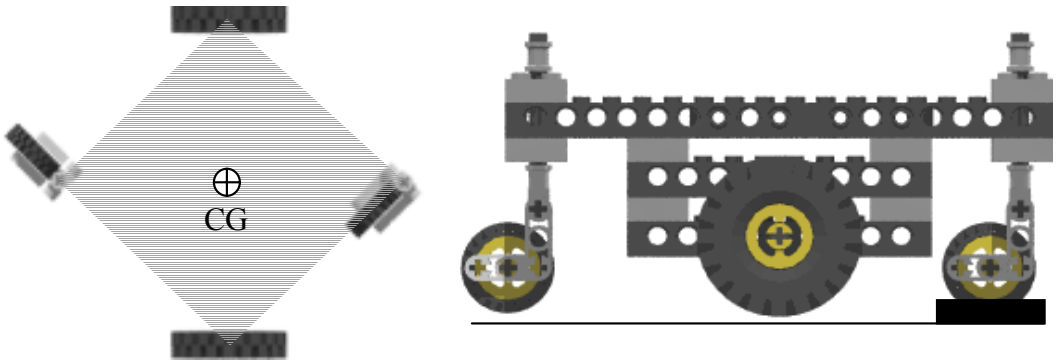


Figure 5-5. Diamond Shaped Wheel Layout

The diamond layout worked great in the lab, gliding along over the smooth flat floors, but it encountered some serious problems out in the real world. To achieve static stability (not fall over when standing still), a vehicle must have at least three wheels, not all on a common axis, touching the ground. If the vehicle has more than three wheels there is no guarantee that all the wheels will be touching the ground at any given moment. A problem with the diamond wheel layout is that when one of the casters drives over a bump it causes one of the drive wheels to lift up into the air. When this happens, the robot turns uncontrollably towards the raised wheel. By the time the caster gets over the bump, the robot is no longer certain of its heading and distance traveled.

A three wheeled triangle configuration fixes this problem, but it loses some of the advantages of the diamond layout. With only three wheels, you are guaranteed to have all three wheels in contact with the ground. Unfortunately, using three wheels also means that the center of gravity cannot be centered over the drive wheel axis, and some of the robotís weight will be supported by the caster. To compensate, robot designers position the CG as close to the drive wheel axis and as far away from the caster as possible. Moving the caster farther away from the drive wheels also helps.

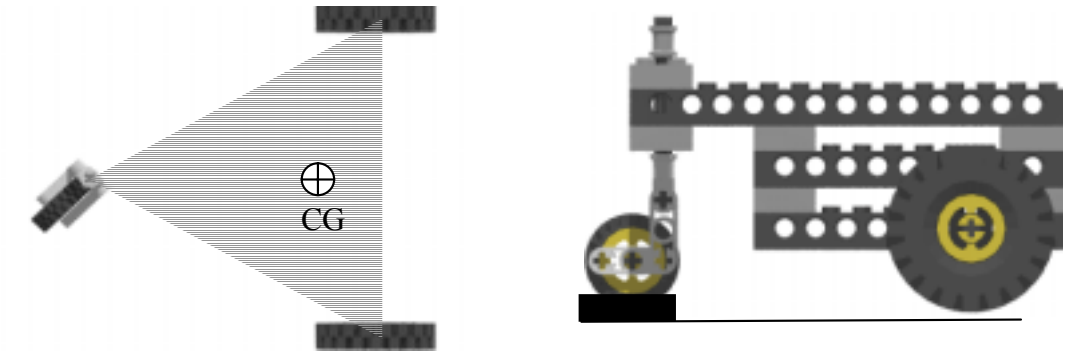


Figure 5-6. Triangle Shaped Wheel Layout

The caster can be placed either in front of or behind the drive wheels. A differential drive robot pivots about a point centered between the two drive wheels. Placing the caster in front of the drive wheels makes the front end of the robot twitchy, very sensitive to steering changes. Placing the caster behind the drive wheels makes the front end more stable.

5.1.3 Steering

A differential drive robot steers by turning the drive wheels at different speeds. Spinning the left wheel faster steers the robot to the right, and spinning the right wheel faster steers the robot to the left. In his article for the Seattle Robotics Society, G. W. Lucas[8] shows that the drive wheels of a differential drive robot follow concentric circular paths while turning. This is shown in the figure below.

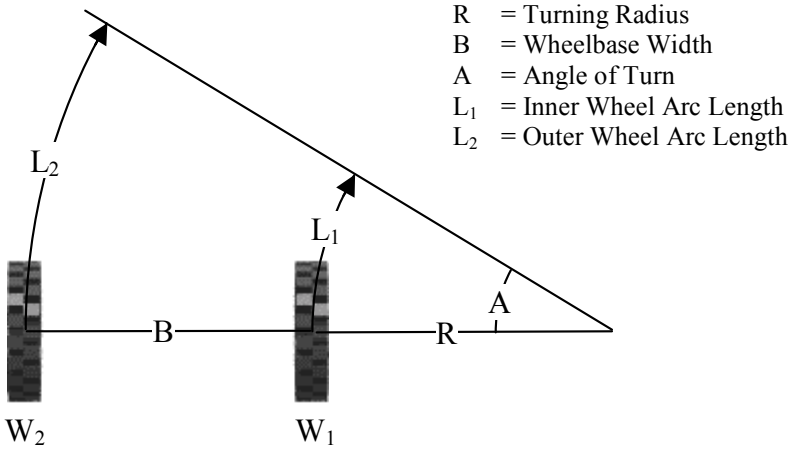


Figure 5-7. Turning Radius is Determined Wheelbase and Relative Speed

If the turn angle (A) is measured in radians then the lengths of arcs L_1 and L_2 can be expressed as:

$$\mathbf{L_1 = A \times R}$$
$$\mathbf{L_2 = A \times (R + B)}$$

[8] ì Using a PID-based Technique For Competitive Odometry and Dead-Reckoningî by G. W. Lucas www.seattlerobotics.org/encoder/200108/using_a_pid.html

With a little algebraic manipulation we can define the turning radius (R) as a function of the wheelbase width (B) and the two arc lengths (L_1 and L_2).

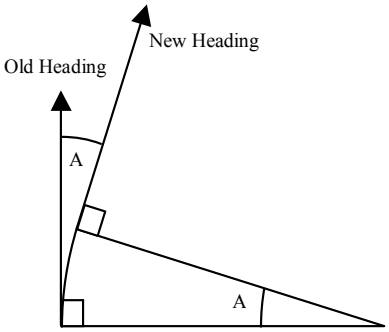
$$A = L_2 / (R + B) = L_1 / R$$
$$R \times L_2 = (R + B) \times L_1$$
$$R \times (L_2 - L_1) = B \times L_1$$
$$R = B \times L_1 / (L_2 - L_1)$$

If we have a differential drive robot with a 4î wide wheelbase, what is the turning radius if the inner wheel is traveling at 18 inches per minute (ipm) and the outer wheel at 24ipm? What would be the heading change if the robot maintained this speed for 10 seconds?

There are several ways to solve this problem, but perhaps the easiest is to calculate the distance each wheel would travel during the 10 seconds, and use these values to calculate the turning radius.

$$L_1 = V_1 \times \text{Time}$$
$$= 18 \text{ ipm} \times 10 \text{ seconds}$$
$$= 18 \text{ ipm} \times 10/60 \text{ minutes}$$
$$= 3î$$
$$L_2 = 24 \text{ ipm} \times 10/60 \text{ minutes}$$
$$= 4î$$
$$R = B \times L_1 / (L_2 - L_1)$$
$$= 4î \times 3î / 4î$$
$$= 3î$$

Now that we know the turning radius, it is easy to calculate the turning angle. The turning angle just happens to be the same as the angular change in the robot heading.



$$\text{Heading Change} = \text{Turning Angle}$$
$$= L_1 / R$$
$$= 3î / (3î / \text{radian})$$
$$= 1 \text{ radian}$$
$$= 57.3 \text{ degrees}$$

Knowing how far each wheel moved, we can calculate the new heading for the robot. Similar equations can calculate how far the robot moved in the North/South and East/West directions. If we wrote the robot control program to continuously calculate heading and position based on the wheel motion, we would be doing odometry. Odometry is a navigation technique where data from wheel position sensors is used to calculate the position and orientation of the robot. Almost all land-based, mobile robots use odometry as part of their navigation system.

5.1.4 Steering Made Easier

The odometry equation discussed in the previous section requires position information from each drive wheel. This makes using odometry for your FLL robot a bit difficult. The rotation sensor is the best device for measuring wheel position, and the challenge set has only one rotation sensor. One solution would be to construct a rotation sensor out of a light sensor or touch sensor. Some teams have done this and gotten it to work pretty well (and scored some major creativity and mechanical design points). Another solution is to rethink the odometry equation, and make it work with only one wheel position sensor.

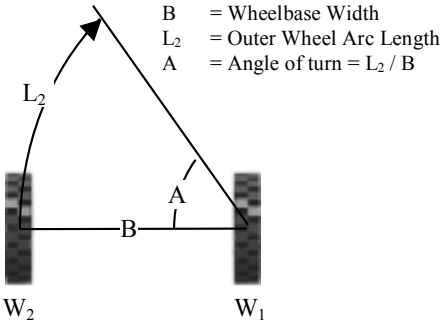


Figure 5-8. Pivoting About a Wheel

Figure 5-8 shows the wheel motion for a zero radius turn. The inner wheel is locked in position, and the outer wheel pivots about it. Because there is no rotation of the inner wheel, it doesnít require a rotation sensor. The turning angle is calculated using the outer wheel motion and the wheelbase.

5.1.5 Straight Line Travel

A problem that many teams have with their differential drive robots is getting them to travel in a straight line. Differential drive robots are extremely sensitive to the relative velocity of the two drive wheels. Even a very small difference in speed will result in the robot traveling in an arc. The tendency to turn can be minimized through good building practices, even weight distribution and choosing well matched motors. But to travel in a perfectly straight line requires some sort of correction mechanism, either mechanical or software.

First, you need to decide if traveling in a straight line is important or not. If your competition strategy is based on making perfect turns and traveling along laser straight paths, your team may be in for some disappointment. The most successful teams build a robot that ì goes straight enoughî and rely on other strategies to compensate for any

navigation errors. These self correcting robots are better able to handle the inevitable variations in starting position or unexpected bumps and slips that occur during the head to head competition.

5.1.5.1.1 Software Solution

Most differential drive robots use encoders (rotation sensors) to measure the position of each drive wheel. To travel in a straight line, the robotís control software continuously monitors the encoder feedback and adjusts the power to the left and right motors to keep the values even. This method is difficult to implement with only one rotation sensor available in the FLL challenge kit. It requires that a second encoder be constructed using the light sensor or a touch sensor.

5.1.5.1.2 Mechanical Solutions

In automotive applications it is often desirable to limit the slip between the left and right drive wheels. A classic example of this is being stuck in the snow. Many times I have seen cars stuck in the snow with one tire spinning wildly on an icy patch and the other tire sitting stationary on dry pavement. The differential supplies the same torque to each wheel, but because one wheel is spinning freely on the ice, the maximum torque to the other wheel is very small.

Many cars in northern climates are equipped with a limited slip differential to solve this problem. With a limited slip differential, if one wheel is spinning faster than the other, the differential removes torque from the faster wheel and supplies it to the slower. This is often done with some sort of clutch that converts a sliding motion into a torque.

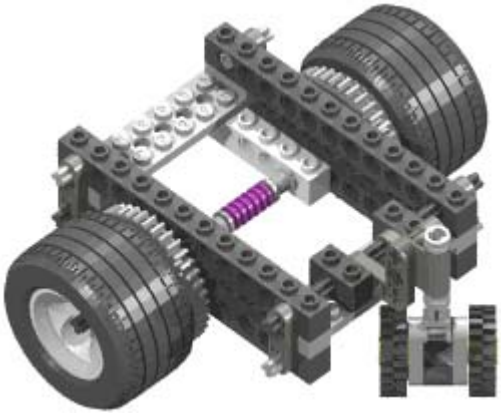


Figure 5-9. Simple LEGO Slip Limiter

The differential drive platform in Figure 5-9 uses a length of ribbed hose as a clutch to limit slip between the two drive wheels. When one wheel spins faster than the other, the axles slip inside the ribbed hose. Friction between the axles and the hose transmits torque from the faster wheel to the slower wheel. The longer the length of axle inside the hose the more torque is transmitted. The robot wonít go perfectly straight, but the amount of turning can be greatly reduced.

Question: What other LEGO parts could be used to design a slip limiting clutch?

Sometimes a limited slip differential isnít enough. To eliminate slip between the left and right drive wheels, some off road vehicles are equipped with a locking differential. A locking differential has a switch that allows it to work like a normal differential, or forces it to behave like a solid axle.

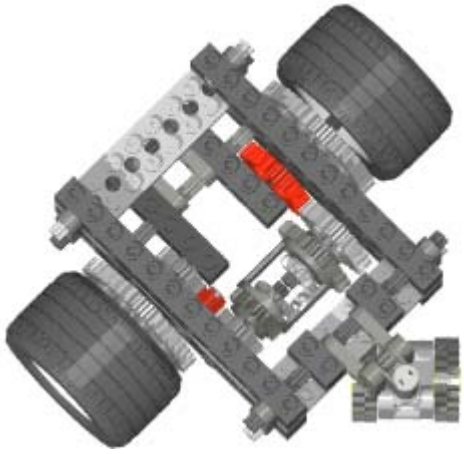


Figure 5-10. A Locking Differential

Way back in the Gears chapter, we talked about how a differential can be used to measure the speed difference between two tires by reversing the rotation of one of the input shafts. This forms the basis for a LEGO locking differential. Preventing the differential in Figure 5-10 from rotating makes the drive train behave as though the left and right wheels were attached to a single axle. Unlocking the differential allows the wheels to spin independently.

Question: Some vehicles have a button or lever in the passenger compartment for locking and unlocking the differential. Can you think of a scheme where the RCX could lock and unlock our LEGO implementation?

5.1.6 Differential Skid

Differential skid, sometimes referred to as skid-steer, is a variation of the differential drive. Itís normally used with tracked vehicles (like tanks), but sometimes with 4 or 6 wheel platforms (like the Bobcat skid loader or Gator ATV) as well. Casters are not used in a differential skid drive; all the wheels are driven. Wheels on the left side are driven by one motor, wheels on the right by another.

Skid steer configurations rely on track or wheel slippage for steering. As a consequence, the wheel position information cannot be used for reliable odometry. For this reason, skid steering is usually reserved for tele-operated (remote controlled) robotic applications. Non-driven measurement wheels can be added to obtain odometry information for autonomous operation.

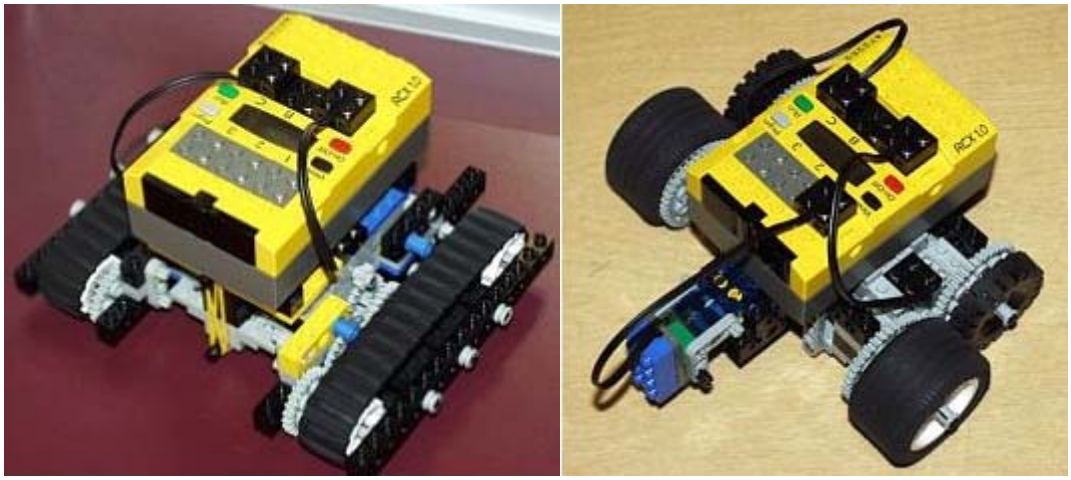


Figure 5-11. A Pair of Differential Skid Robots

Many differential skid robots have competed in Minnesota FLL competitions over the past few years, and several have done quite well. The successful teams developed strategies that maximized their robotsí strengths and minimized their robotsí weaknesses. Skid steer robots are good climbers and can handle rougher terrain than most other robot designs. They are as maneuverable as a differential drive robot, but less prone to unwanted differential steer. And poor odometry data isnít a problem if your robot uses line following or some other landmark-based strategy for navigation.

5.2 Steering drive

A steering drive is the familiar configuration used in automobiles. In a steering drive, one motor is used for locomotion and a second motor for steering. Decoupling steering from forward motion makes a steering drive easier to control than a differential drive where the velocity of each drive wheel must always be carefully measured and controlled. Front wheeled steering provides accurate odometry while supporting the traction and ground clearance needs of all-terrain operation. This makes it a popular choice for outdoor autonomous vehicles.

In general, a steering drive robot is more difficult to build than a differential drive robot. It needs a differential to transfer the power of the drive motor to the wheels. Extra gearing is usually required to amplify the torque output of the single drive motor. The steering mechanism can also be difficult to design using the parts available in the RIS kit. Luckily, there are many Technic steering drive examples on the web. A quick search on LUGNET revealed over 100 different steering drive vehicles.

Figure 5-12 shows two different steering drive robots. The robot on the left uses crown gears and spur gears to pivot the front axles. The robot on the right has rack and pinion steering just like most cars on the road today. The robot on the right uses a special gear rack that is available in many Technic vehicle sets, but a similar mechanism could be built using the gear racks supplied in the RIS kit.

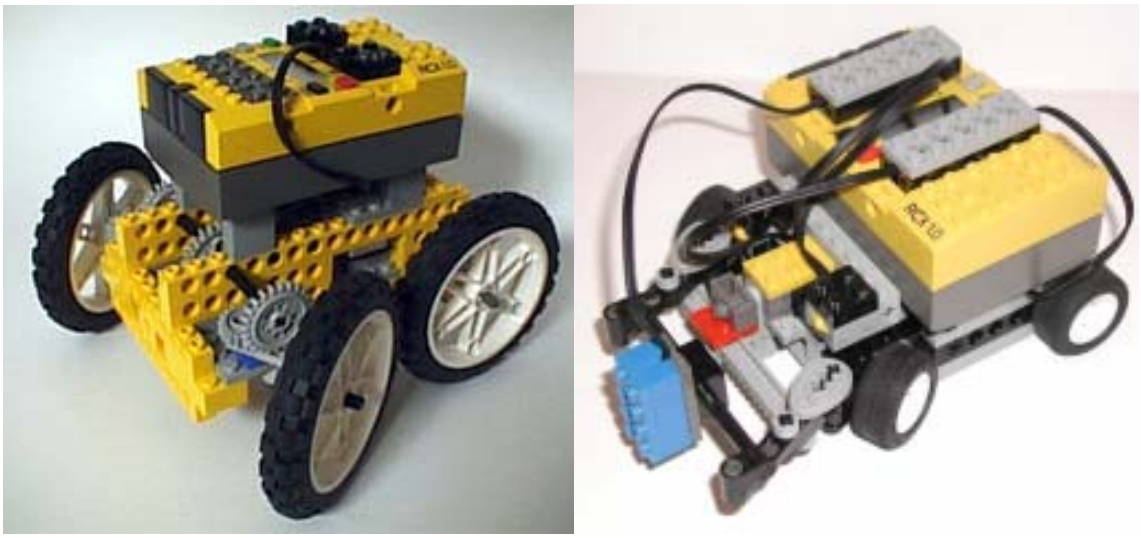


Figure 5-12. Two steered wheel robots

5.2.1 Turning

A steering drive robot has a non-zero turning radius. This means that unlike a differential drive robot, a steering drive robot must translate (move forward, backwards, and/or sideways) while turning. Such systems are said to be non-holonomic. In a non-holonomic robot, one or more of the degrees-of-freedom (heading, position) cannot be controlled independently. This makes path planning more difficult because each heading change will have an accompanying position change. Non-holonomic vehicles can be particularly difficult to navigate in tight spaces. Remember the last time you parallel parked?

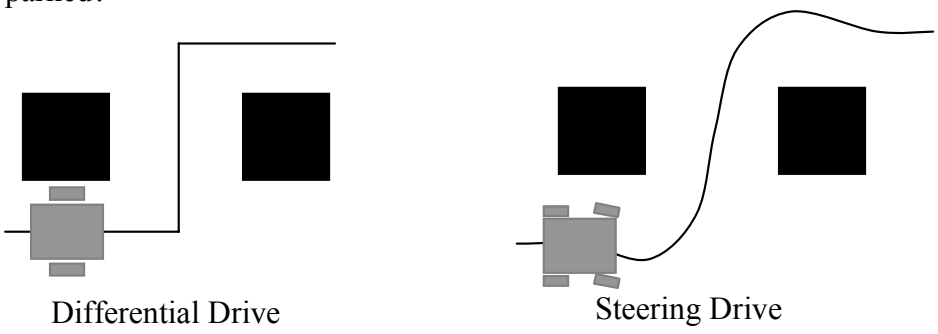


Figure 5-13. Path Planning is Harder for Non-Holonomic Robots

The only odometry information I could find on the web was for steering platforms with Ackerman correction. Used in almost every vehicle you see on the road, Ackerman steering correction is a modified steering geometry that minimizes wheel skid when turning. During steering, the four wheels of a steering drive travel along four concentric arcs. As seen in Figure 5-14, the front wheels of a vehicle with Ackerman correction are oriented such that the extended axis of the wheels passes through the center of these arcs. This requires the inner wheel to turn at a sharper angle than the outer wheel.

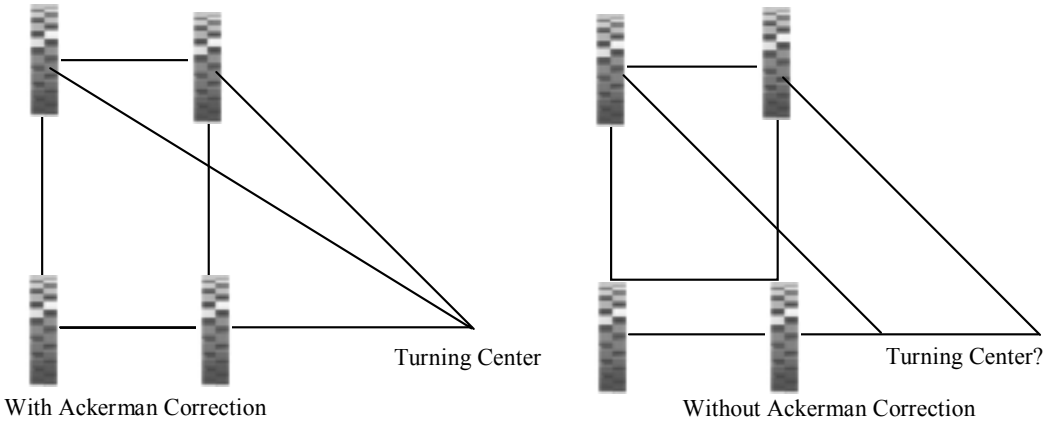


Figure 5-14. Ackerman Steering Minimizes Geometry Induced Wheel Skid

Without Ackerman correction, the front wheels both turn the same amount when steering. This results in them trying to travel along arcs with different centers. As the vehicle turns, one or both of the front wheels are forced to skid sideways, generating large forces on the steering components and producing excessive wear on the tires. More important for robotics applications, is the fact that you canít calculate solutions for the odometry equations because the turning center cannot be absolutely determined.

Itís possible but difficult to build a LEGO steering drive with Ackerman correction. I found some web pages with calculators for designing the components and geometry and a few books devoted to the subject. Unfortunately, the solutions require using specifically sized components that often donít map well to my available LEGO pieces.

A simpler solution is to use only one steered wheel. Three wheeled steering drives are most common in specialized niche markets like golf or meter reader carts. Their main disadvantage, a tendency to tip over when turning at high speed, is not a big problem in these applications. However, a three wheeled platformís simpler construction and odometry makes it a good choice for a LEGO robot.

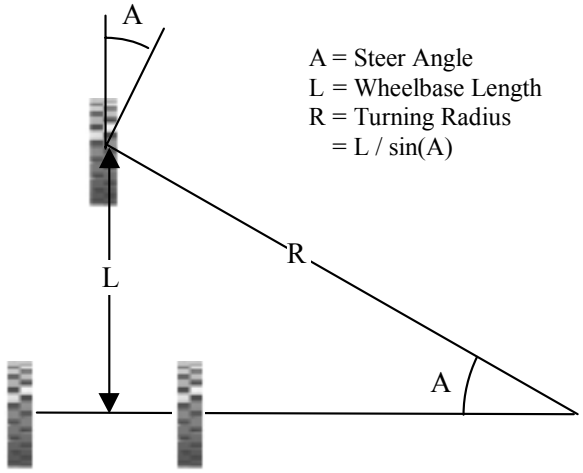


Figure 5-15. Turning Radius is Determined by Wheel Base and Steering Angle

As shown in Figure 5-15, the turning radius of a steering drive is determined by the length of the wheelbase and the steer angle of the front wheel(s). This is why long trucks have a large turning radius and small sports cars a tight turning radius.

Letís say we have a steering drive robot with a 6î long wheelbase and a maximum steer angle of 30 degrees. What is the minimum turning radius for the front wheel, and how far would the front wheel travel while completing a 45 degree turn?

Turning Radius = L / sin(A)
= 6î / sin(30 degrees)
= 12î

Travel Distance = Radius x Angle (in radians)
= 12î x 45 degrees x (Pi radians / 180 degrees)
= 12î x Pi / 4
= 9.42î

5.2.2 Tricycle Drive

A tricycle drive is a steered drive platform with a single driven front wheel and two passive rear wheels (or vice versa). Though it may look very similar to a three wheeled steering drive platform, a tricycle drive has some unique characteristics that make it preferable for most robotic applications. The most important of these is its ability to turn in place.

Figure 5-16. Tricycle Drive (Left) and Steering Drive (Right) Robots Look Similar

As discussed in the previous section, the minimum turning radius for a steering drive is determined by the maximum steer angle and the length of the wheelbase, R = L / sin(A). If the maximum steer angle was 90 degrees, then the turning radius would be equal to the wheelbase length. The robot could pivot about a point midway between the two rear wheels.

Unfortunately, a steering drive will not turn if the steer angle is 90 degrees. If the vehicle moves at all, the front wheel(s) skid sideways instead of rolling. This is because the drive

forces from the rear wheels are perpendicular to the direction of travel of the front wheel. All of the force is directed at pushing the wheel sideways, and none at pushing it forwards. Sideways wheel slip is noticeable even at steer angles significantly less than 90 degrees. About the best you can hope for is to place the turning center near the outside of the drive wheels

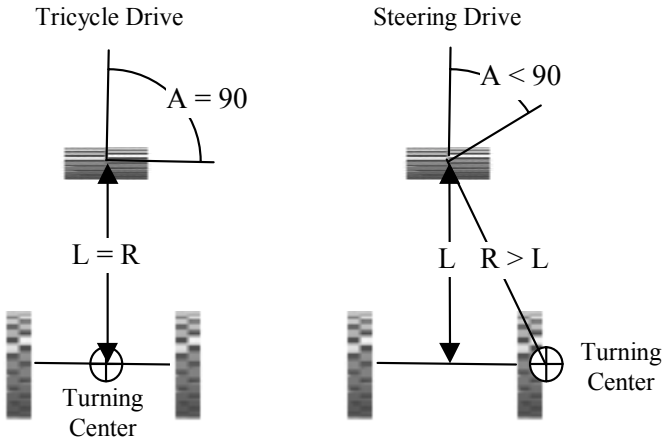


Figure 5-17. Any Steer Angle is Possible with a Tricycle Drive

A tricycle drive robot has the drive motor attached to the steered wheel, allowing it to operate at any steer angle. This gives it maneuverability that rivals that of differential drive and skid steering robots.