Turning Tutorial

Ends:

- Improve Student Understanding of Circular Relationships.
- Enable Students to logically develop an approach to converting a desired robot heading change into software that accurately implements the change

Means:

- Demonstrate math to support student comprehension of the relationships between the two major circles associated with a simple Lego NXT robot
- Provide Students sufficient background to develop their own turning programs.
- Provide example programs
- Culminate Student comprehension by programming the robots to move in any regular polygon clockwise or counterclockwise as directed by the instructor.

Tools:

- Acrobat file titled "Making Precise Turns - Understanding the Relationships between the Robot and its Wheels" which includes:
  - attached MS Office and Open Office file of same presentation
  - attached MS Office and Open Office file of spreadsheet implementing the relationships
  - RoboLab screenshots of straight line and turning programs.
  - RoboLab screenshot of a regular polygon shape program using subroutines developed from the straight line and turn program.

Discussion:

Most robot competitions I've participated in seem to require the robots to be driven to a specific location on the robot course, do something there, and then return to a declared home base for the next assignment.

After participating with four different teams and three competitions, this seems the one place were a strong mentoring hand can have a good influence, laying the groundwork for other approaches to controlling robot movement. And for competitions that can rely on odometry to get the robot from one place to another, getting the students through this early step, should enable them to accomplish more with less direct mentor intervention.

Regardless of more sophisticated methods to determine the robot's location relative to another place on the board, these seem to be necessary baseline tools to get the robot from one place to another.

I've only included a set of screenshots from RoboLab. If there's interest and time, I will explore developing parallel screenshots for NXT-G and LabView.

The robot depicted is based on a slightly modified Domabotics model.

Comments and criticism welcome.

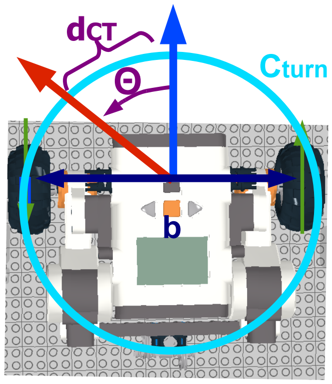
All my best,
Craig Shelden
craig@sheldenrobotics.com

Shelden Robotics

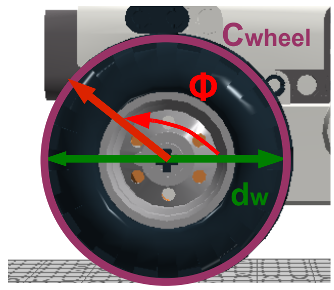
One Page Handout

Making Precise Turns:
Understanding the Relationships between the Robot and its Wheels

Top View



Side View



<u>Terms Defined</u>

b	**Robot Wheelbase**
C_{turn}	**Robot Circumference of turn**
	$C_{turn} = b \times \pi$
Θ	**Desired Turn Angle**
d_{CT}	**Distance along C_{turn}**
d_w	**Wheel Diameter**
C_{wheel}	**Wheel Circumference**
	$C_{wheel} = d_w \times \pi$
Φ	**Motor Rotation**
360°	degrees around a circle

We want the robot to turn a set number of degrees (Θ) by turning the power wheels in opposite directions. Θ can also be expressed in terms of its fractional value of the whole circumference and it is also equal to the distance along the circumference we want to turn the robot.

$$(\Theta/360) = (d_{CT} / C_{turn})$$

And we can solve this for the Distance along the circumference to turn the wheels.

$$d_{CT} = (\Theta/360°) \times (C_{turn})$$

We want to be able to program this in terms of wheel rotations or degrees.

Recognize that the distance along the Robot's Turning Circumference (d_{CT}) is the same distance the robot's wheels must turn along the Circumference of each Wheel (C_{wheel}) to make the turn.

This leads to the following relationship:

$$\text{Number of Rotations to Program} = (d_{CT}) / (C_{wheel}) \quad \text{in NXT-G}$$

$$\text{Number of Degrees to Program} = (360°) \times (d_{CT}) / (C_{wheel}) \quad \text{in NXT-G or RoboLab}$$

And if we substitute the terms that make d_{CT} into the above equations, they simplify rather nicely.

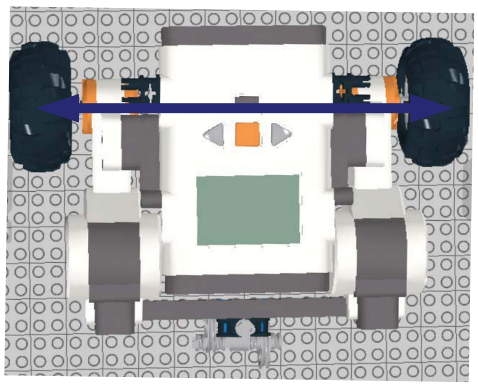
$$\text{Number of Rotations to Program} = (C_{turn}) / (C_{wheel}) \times (\Theta) / (360°) \quad \text{in NXT-G}$$

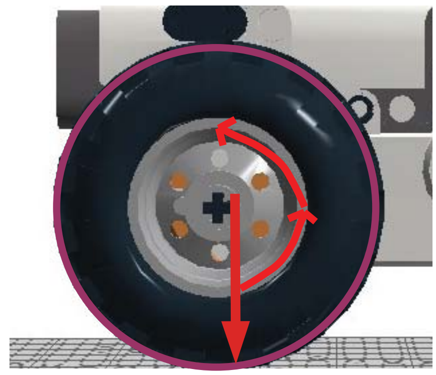
$$\text{Number of Degrees to Program} = (C_{turn}) / (C_{wheel}) \times (\Theta) \quad \text{in NXT-G or RoboLab}$$

Detailed Tutorial Slides

Making Precise Turns

Understanding the Relationships between the Robot and its Wheels



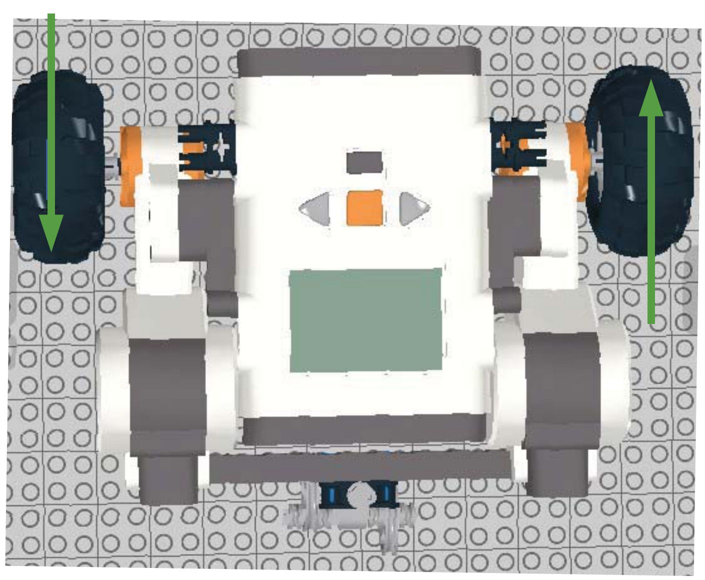


Turn Concept

We're going to turn the robot by driving one wheel backward and one wheel forward.

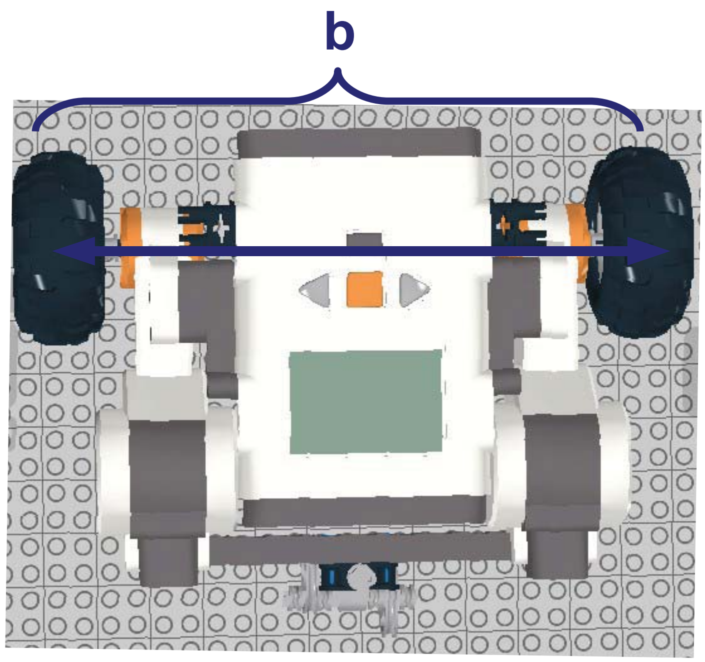
Telling the robot to turn each motor "some amount" forces the wheel to travel along C_{turn}.

The distance along C_{turn} – when moved ahead on one side and backwards on the other, turns the robot.



Wheelbase

The distance between Robot wheel centers is called the **Wheelbase** (**b**)



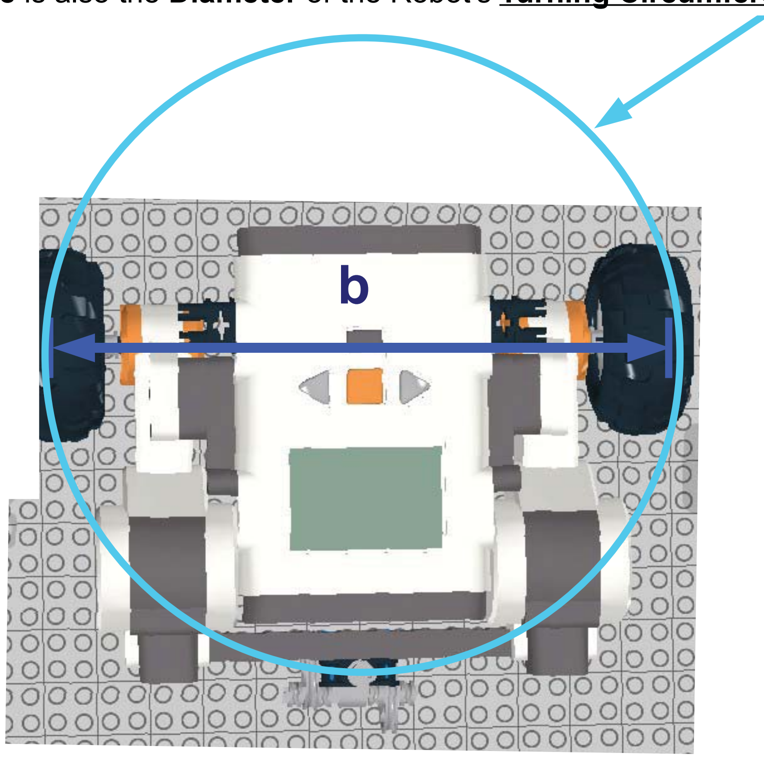
Turning Circle or Turning Circumference

The **Wheelbase** is also the **Diameter** of the Robot's **Turning Circumference** (**Cturn**)

The turning circumference defines how tight a turn the robot can make.

The length of the Turning Circumference is defined as

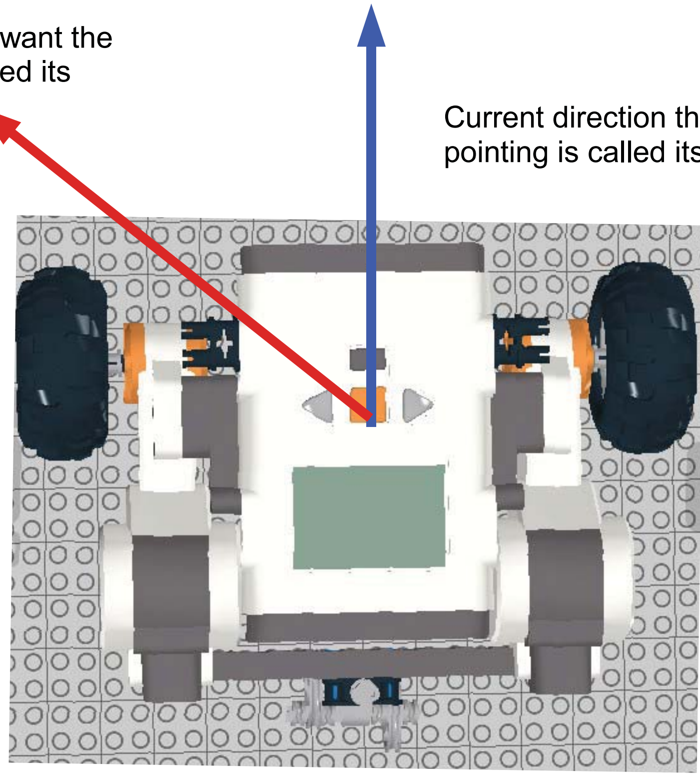
$$C_{turn} = b \times \pi$$



But we need to know how far to turn it: Robot Heading

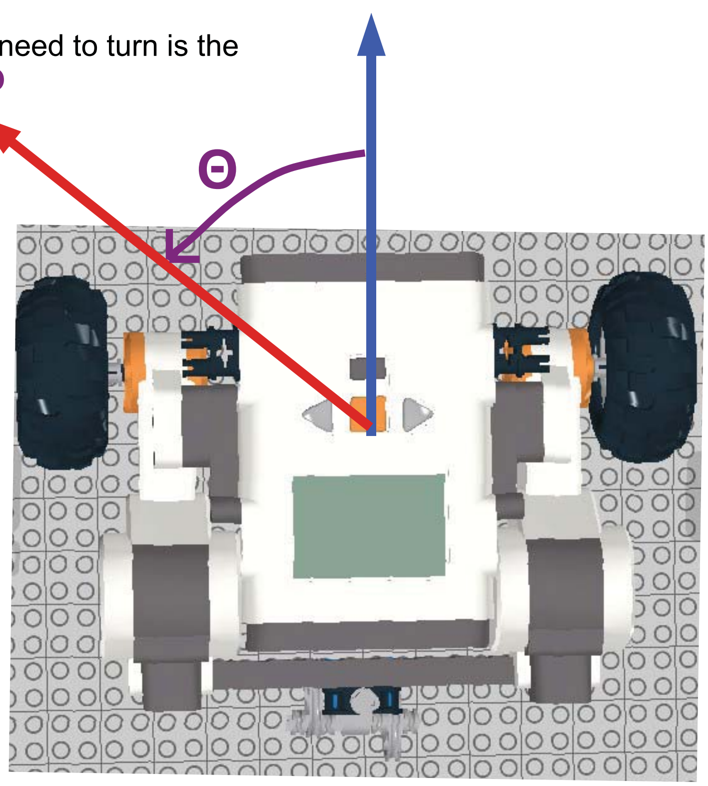
Desired direction we want the robot to turn to is called its **New HEADING**

Current direction the robot is pointing is called its **HEADING**

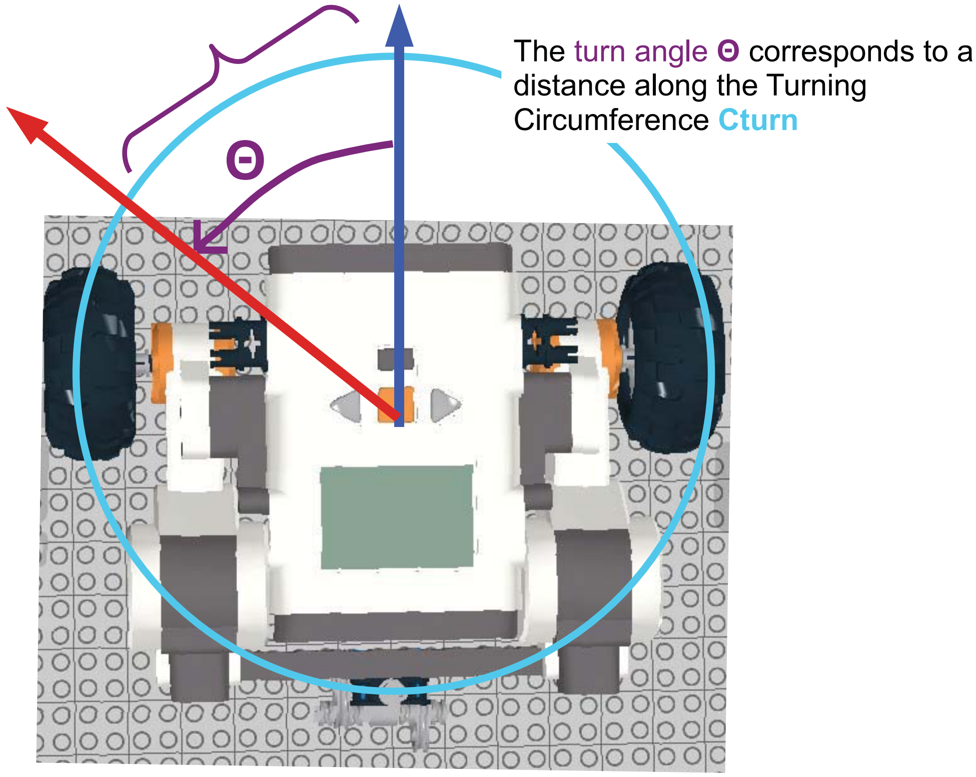


Now for Turn Angle (Θ) Robot Heading

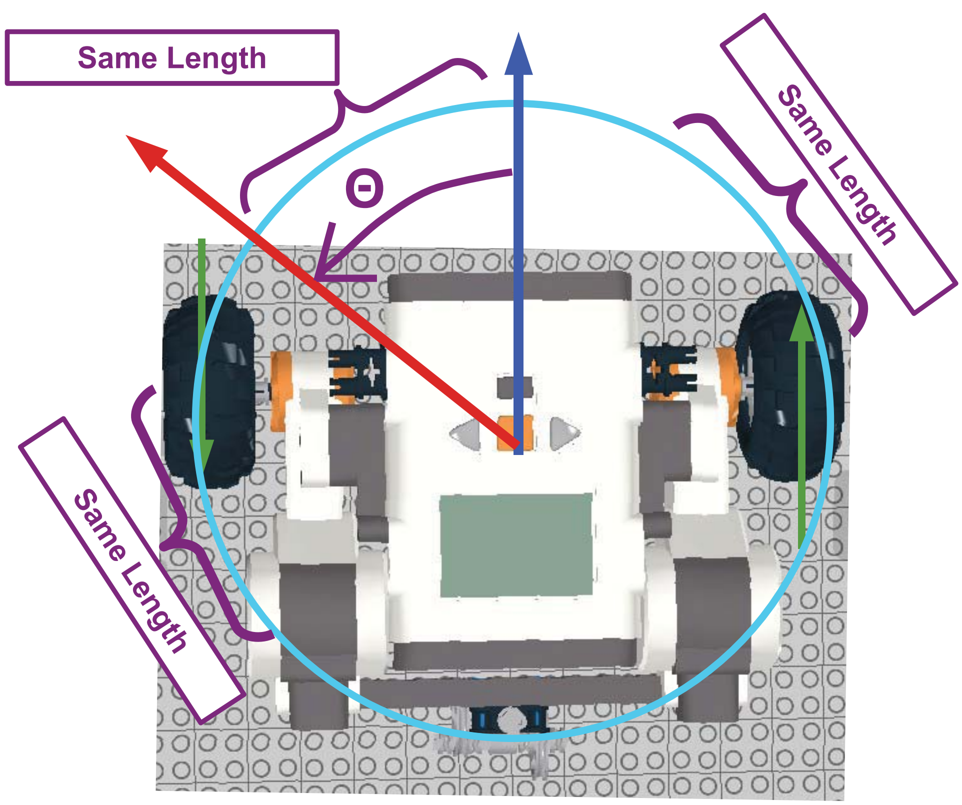
How far we need to turn is the turn angle **Θ**



Now for Turn Angle (Θ)
Defining Robot Heading

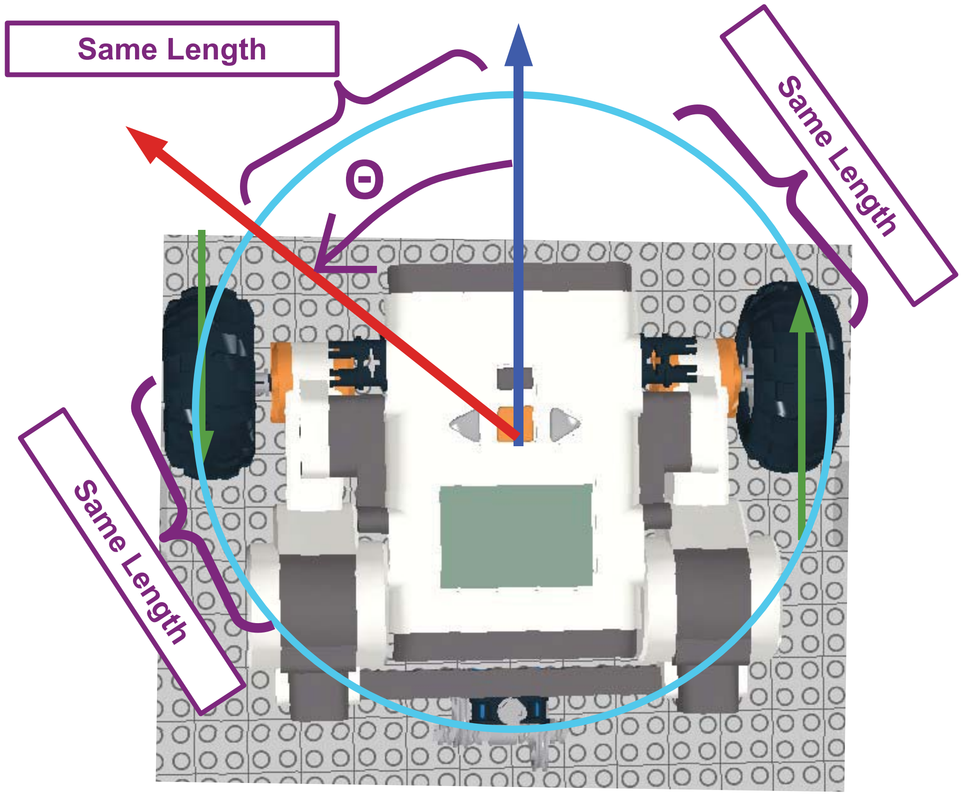


Turn Concept



All the purple brackets { are the **same length.**

Giving “Same Length” a Name → dCT



All the purple brackets { are the **same length.**

Figuring Out How Far to Turn

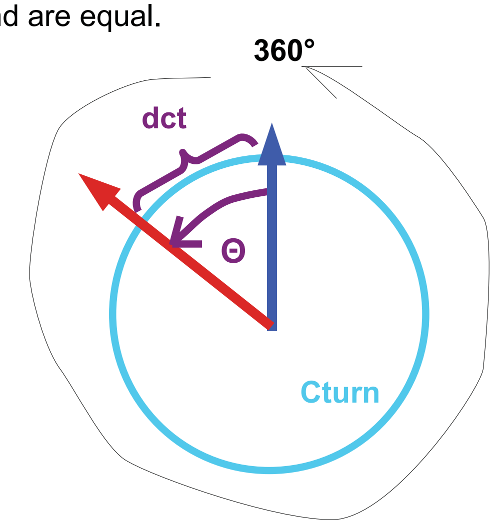
dCT or “**Distance along Cturn**” can be expressed as a fraction of the Robot's **Turning Circumference** (**Cturn**)

And

The degrees of heading change we want to make (Θ) can be expressed as a fraction of the # of degrees in a circle (360°).

These two fractions both describe the same thing – and are equal.

$$\frac{dCT}{Cturn} = \frac{\Theta}{360°}$$



Figuring Out How Far to Turn

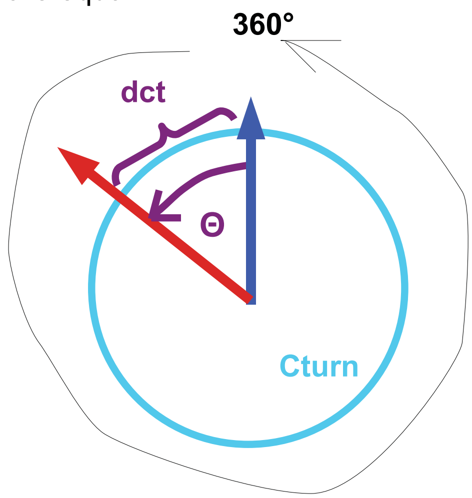
dCT or "**Distance along Cturn**" can be expressed as a fraction of the Robot's **<u>Turning Circumference</u>** (**Cturn**)

And

The degrees of heading change we want to make (**Θ**) can be expressed as a fraction of the # of degrees in a circle (360°).

These two fractions both describe the same thing – and are equal.

$$\frac{dCT}{Cturn} = \frac{\Theta}{360°}$$



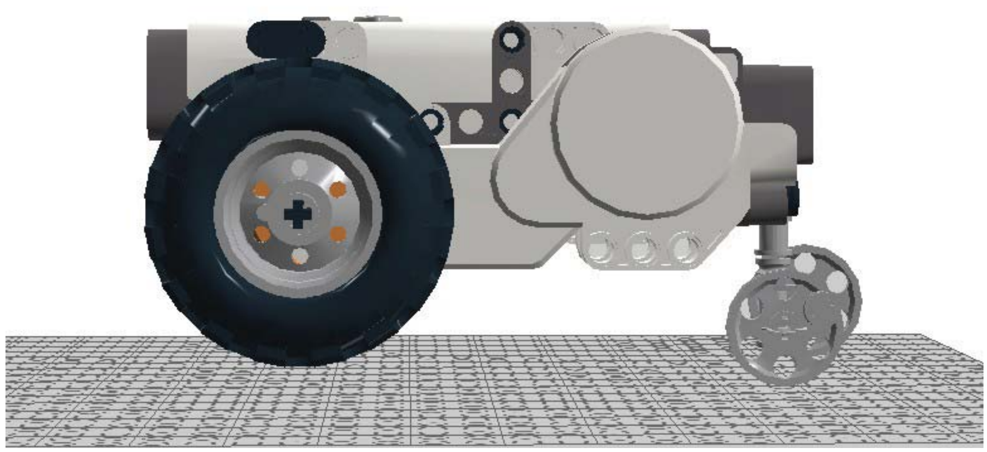
And we can solve for **dCT**

$$dCT = Cturn \times \frac{\Theta}{360°}$$

But we need to tell the motors how much to turn So there are a few more things to look at

Remember, the motors don't know how far they've traveled, only the degrees or rotations they've made.

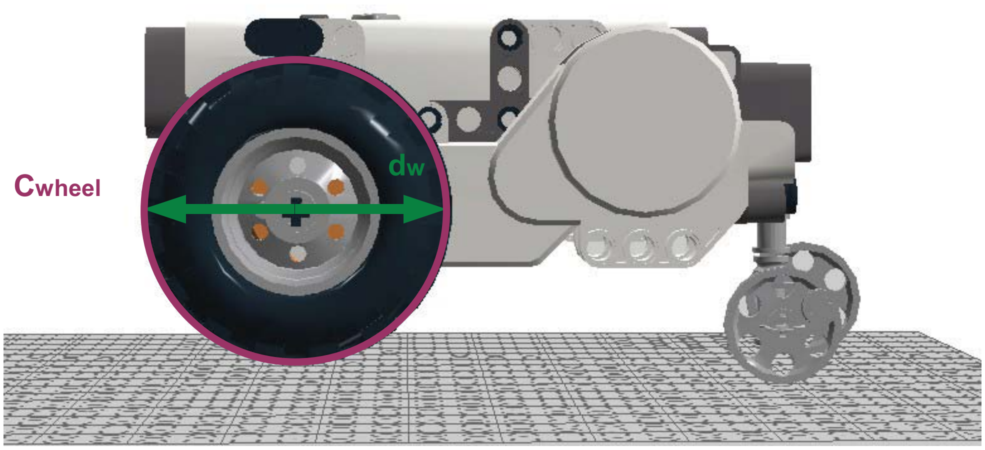
How far they travel – "some amount" – is up to the Programmer.



Motor Rotation – Just a few more terms

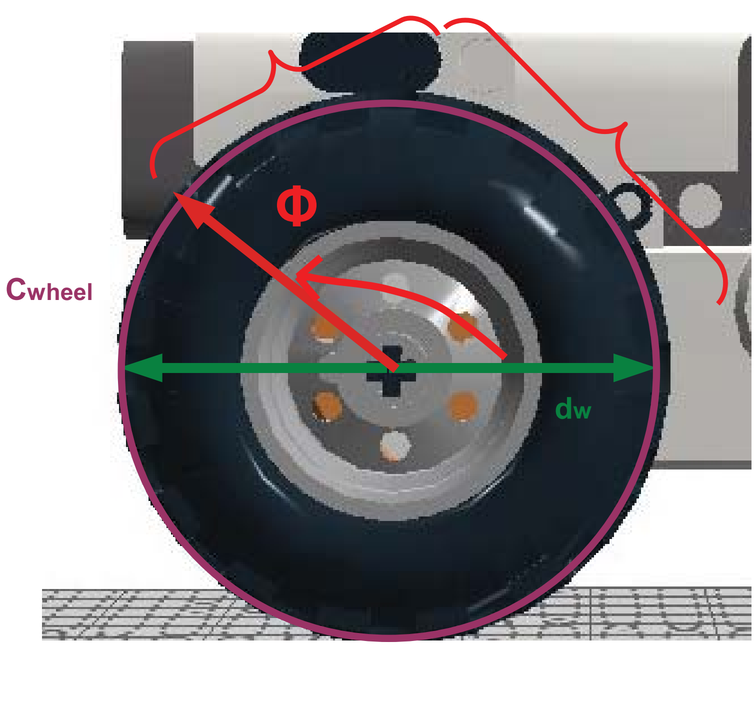
The **Diameter of the Wheel (d_w)** and **Wheel Circumference (C_{wheel})** determine how far the wheels of the robot move for a given number of degrees.

$$C_{wheel} = d_w \times \pi$$



Motor Rotation – Finally something we can Program!

We tell the motors how much to turn each wheel using either degrees (RoboLab & NXT-G) or Rotations (NXT-G). **Wheel Turn Angle (Φ) is** how far we tell the motor to turn the wheel.



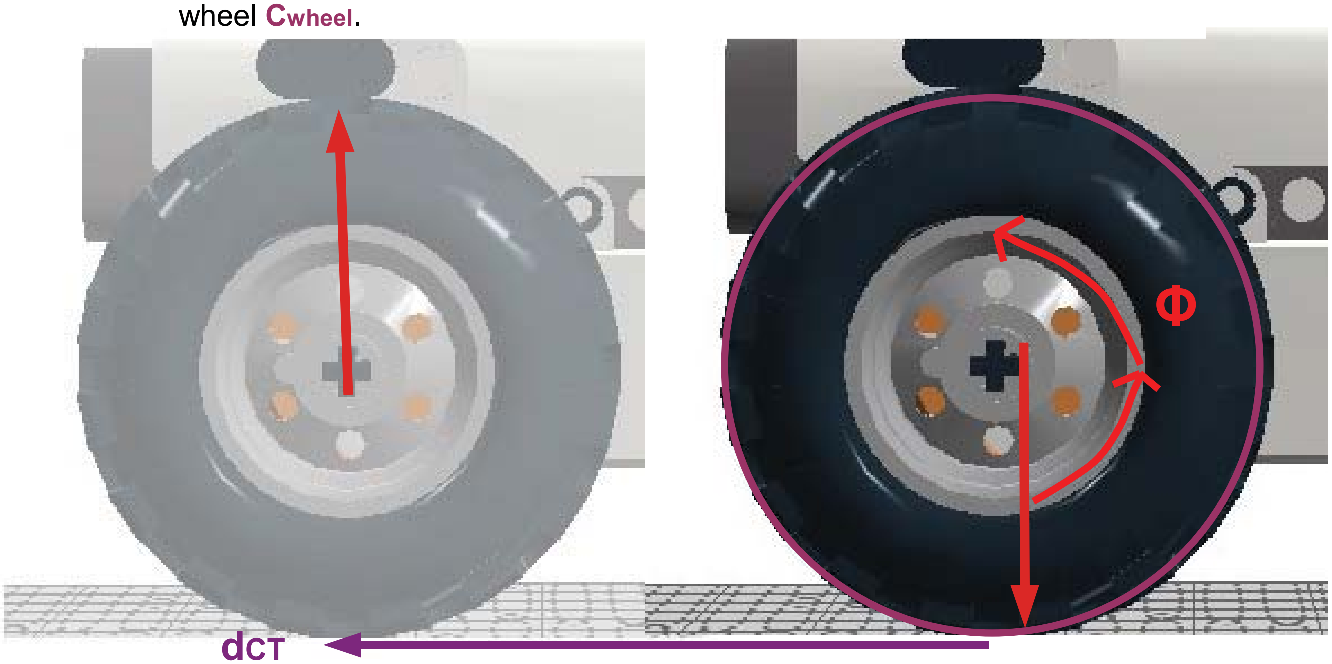
Telling the robot to turn the motor by **Φ** forces the wheel to travel along **C_{turn}.**

The distance along **C_{turn}** – when moved ahead on one side and backwards on the other, turns the robot.

Motor Rotation – Finally something we can Program!

Let's look at how we can express d_{CT} in terms of the wheels and the Wheels' turn angles.

In the picture below we're moving the robot from right to left by rotating the wheel an angle Φ that corresponds to d_{CT} along the circumference of the wheel C_{wheel}.

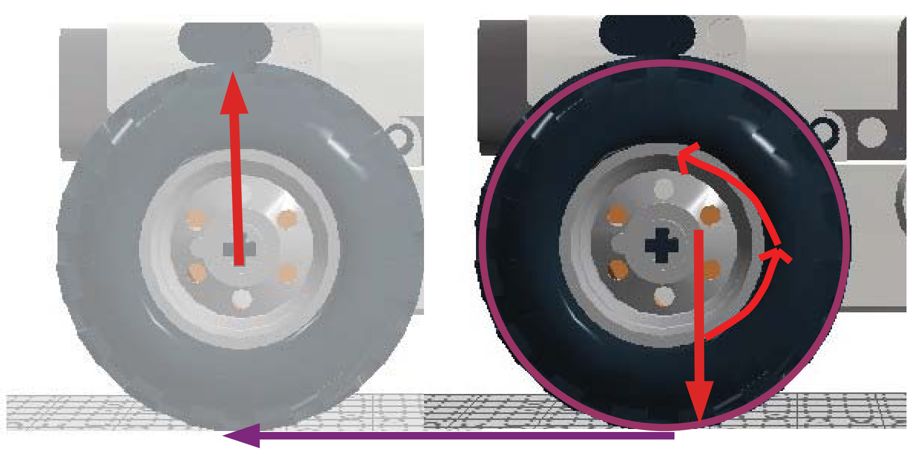


Motor Rotation – Finally something we can Program!

And now we can see that d_{CT} is a fraction of C_{wheel} as well as a fraction of C_{turn}.

And how far to tell the program to move the wheel:

$$\textbf{\# of Wheel Motor Rotations} = \frac{d_{CT}}{C_{wheel}} \qquad \text{For NXT-G}$$

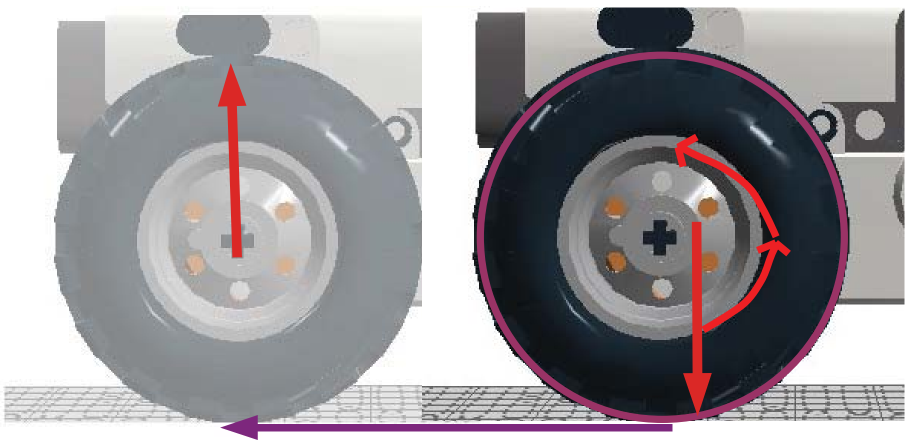


Motor Rotation – Finally something we can Program!

And now we can see that **dCT** is a fraction of **Cwheel** as well as a fraction of **Cturn.**

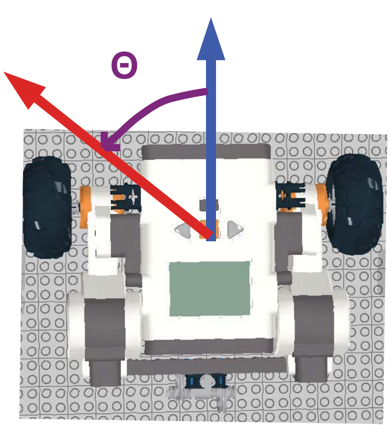
And how far to tell the program to move the wheel:

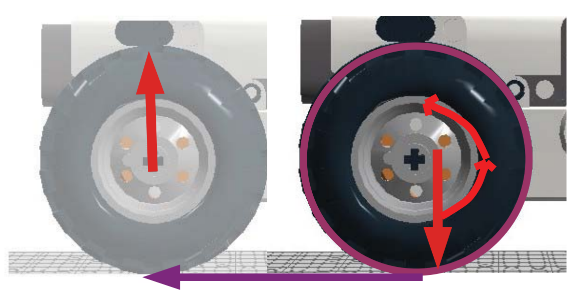
$$\textbf{\# of Wheel Motor Degrees} = 360° \times \frac{d_{CT}}{C_{wheel}}$$ For NXT-G and Robolab



Motor Rotation – Don't be Surprised

For large Turn Angles **Θ** or small wheels, **dCT** is sometimes a fraction that is bigger than one.



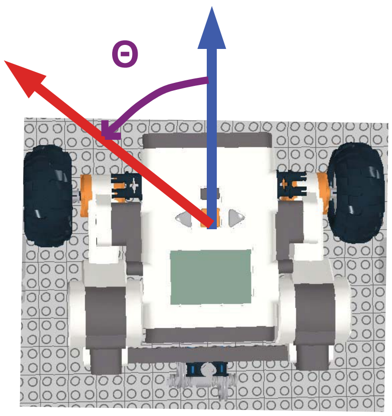


Summary

Based on our mission, we know how many degrees we need to turn the robot, or our desired turn angle **Θ.**

And **Θ** corresponds to a fraction (**dCT**) of our robot's turning circumference (**Cturn**):

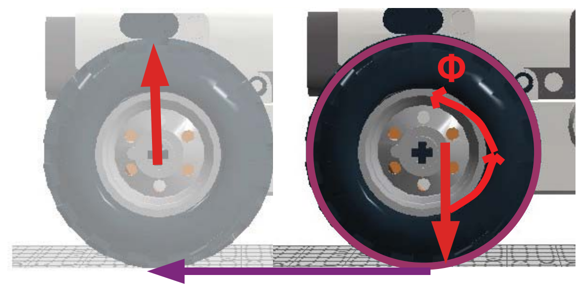
$$\textbf{dCT} = \textbf{Cturn} \times \frac{\Theta}{360°}$$



And **dCT** is **also** a fraction of **Cwheel**

$$\textbf{\# of Wheel Motor Rotations} = \frac{\textbf{dCT}}{\textbf{Cwheel}}$$

$$\textbf{\# of Wheel Motor Degrees} = 360° \times \frac{\textbf{dCT}}{\textbf{Cwheel}}$$



Summary – 2 Combining terms

Based on our mission, we know how many degrees we need to turn the robot, or our desired turn angle **Θ.**

And **Θ** corresponds to a fraction (**dCT**) of our robot's turning circumference (**Cturn**):

And **dCT** is **also** a fraction of **Cwheel**

Substituting **dCT**'s factors into the # of Rotations or Degrees has some interesting results.

$$\textbf{dCT} = \textbf{Cturn} \times \frac{\Theta}{360°}$$

$$\textbf{\# of Wheel Motor Rotations} = \frac{\textbf{dCT}}{\textbf{Cwheel}} = \frac{\textbf{Cturn}}{\textbf{Cwheel}} \times \frac{\Theta}{360°}$$

$$\textbf{\# of Wheel Motor Degrees} = 360° \times \frac{\textbf{dCT}}{\textbf{Cwheel}} = 360° \times \frac{\textbf{Cturn}}{\textbf{Cwheel}} \times \frac{\Theta}{360°}$$

Summary – 2 Combining terms

Based on our mission, we know how many degrees we need to turn the robot, or our desired turn angle **Θ.**

And **Θ** corresponds to a fraction (**dCT**) of our robot's turning circumference (**Cturn**):

And **dCT** is **also** a fraction of **Cwheel**

Substituting **dCT**'s factors into the # of Rotations or Degrees has some interesting results that make all the math EASIER.

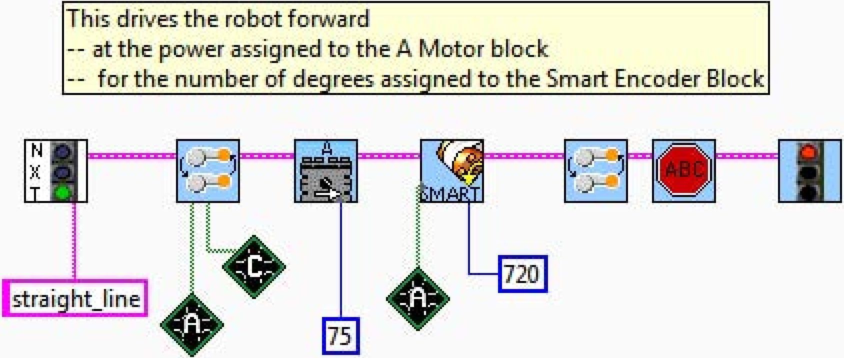
$$dCT = Cturn \times \frac{\Theta}{360°}$$

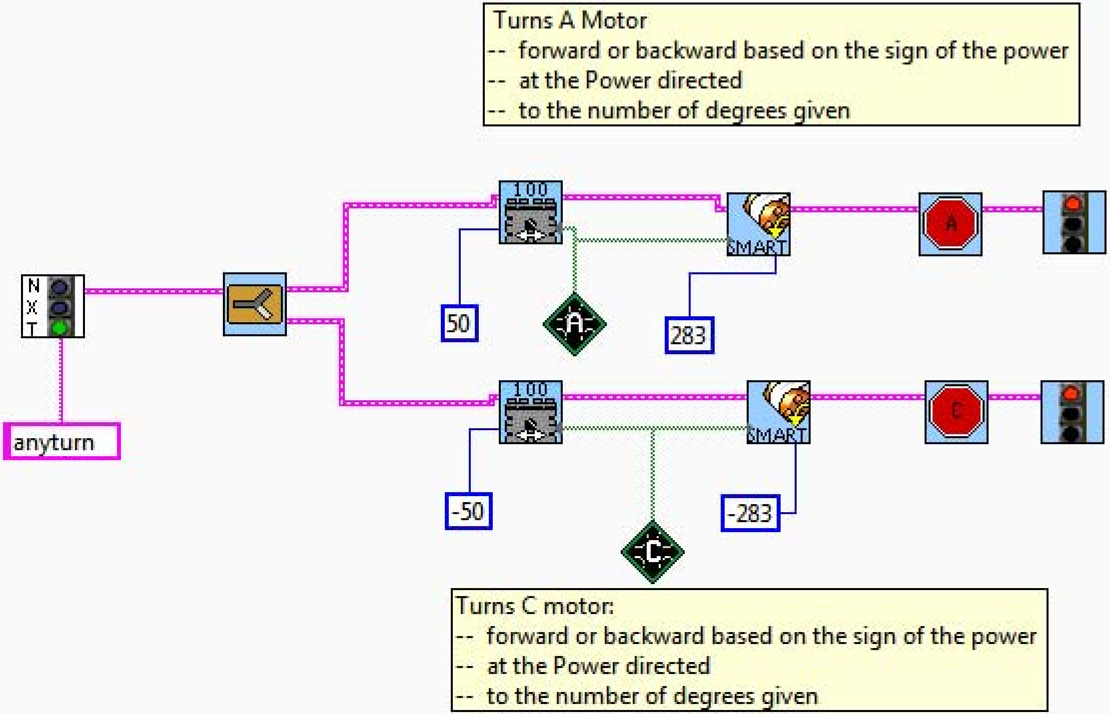
$$\text{\# of Wheel Motor Rotations} = \frac{dCT}{Cwheel} = \frac{Cturn}{Cwheel} \times \frac{\Theta}{360°}$$

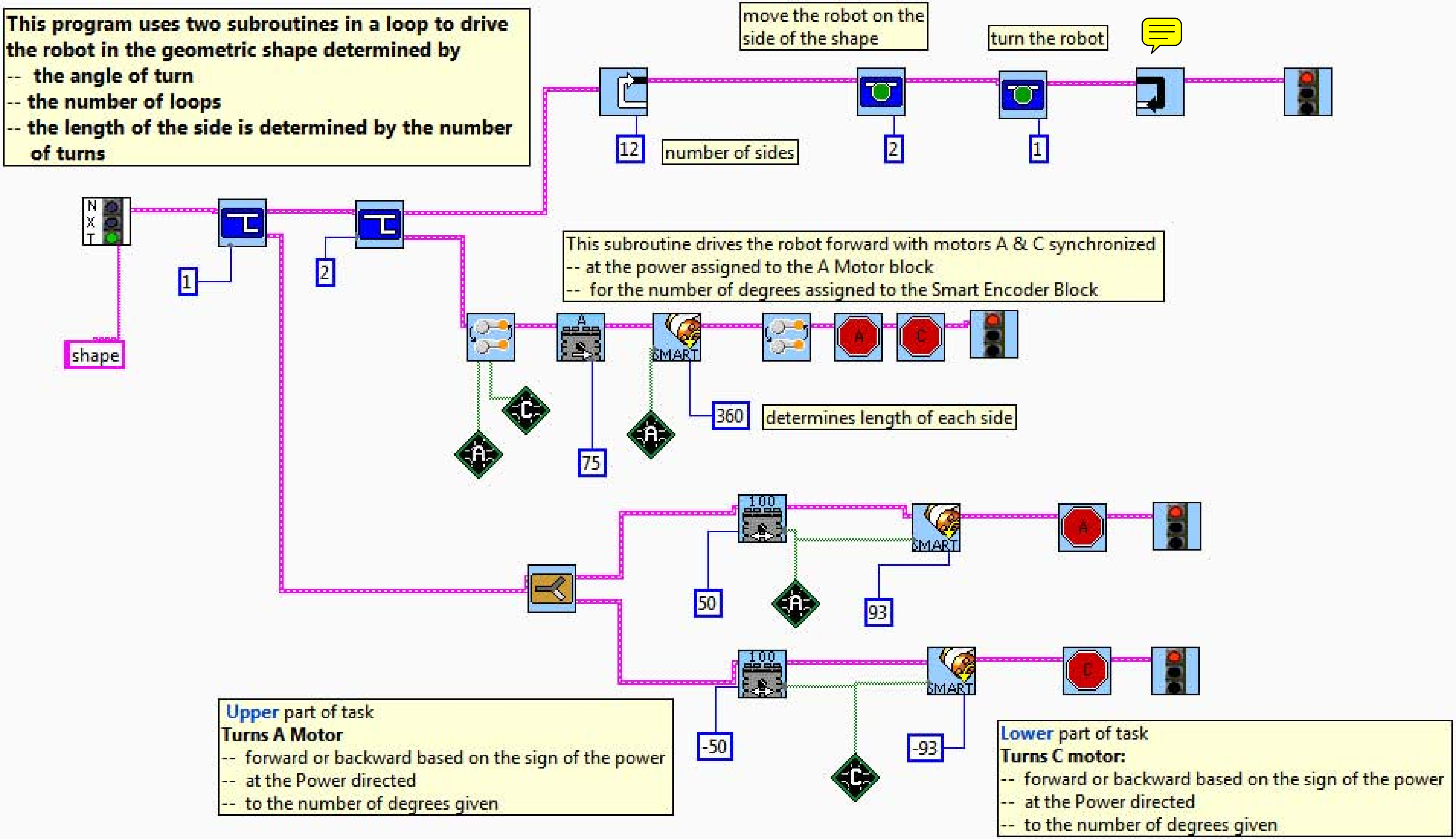
$$\text{\# of Wheel Motor Degrees} = 360° \times \frac{dCT}{Cwheel} = \cancel{360°} \times \frac{Cturn}{Cwheel} \times \frac{\Theta}{\cancel{360°}}$$

$$\text{\# of Wheel Motor Degrees} = \frac{Cturn}{Cwheel} \times \Theta$$

RoboLab Screenshots


This drives the robot forward
-- at the power assigned to the A Motor block
-- for the number of degrees assigned to the Smart Encoder Block
N
X
T
straight_line
A
C
A
75
SMART
A
720
ABC


Turns A Motor
-- forward or backward based on the sign of the power
-- at the Power directed
-- to the number of degrees given
N
X
T
anyturn
100
50
A
SMART
283
100
-50
C
SMART
-283
Turns C motor:
-- forward or backward based on the sign of the power
-- at the Power directed
-- to the number of degrees given


This program uses two subroutines in a loop to drive the robot in the geometric shape determined by
-- the angle of turn
-- the number of loops
-- the length of the side is determined by the number of turns
move the robot on the side of the shape
turn the robot
12
number of sides
2
1
NXT
shape
1
2
This subroutine drives the robot forward with motors A & C synchronized
-- at the power assigned to the A Motor block
-- for the number of degrees assigned to the Smart Encoder Block
A
SMART
A
C
C
A
75
A
360
determines length of each side
100
SMART
A
50
A
93
100
SMART
C
-50
C
-93
Upper part of task
Turns A Motor
-- forward or backward based on the sign of the power
-- at the Power directed
-- to the number of degrees given
Lower part of task
Turns C motor:
-- forward or backward based on the sign of the power
-- at the Power directed
-- to the number of degrees given

Ungeared Robot Turn Calculator

b	**7**	Robot Wheelbase
Cturn	21.99	Turn Circumference
Cwheel	**7.1**	Wheel Circumference

Angle to turn	# of Degrees
0	0
5	15.49
10	30.97
15	46.46
20	61.95
25	77.43
30	92.92
35	108.41
40	123.89
45	139.38
50	154.87
55	170.35
60	185.84
65	201.33
70	216.81
75	232.3
80	247.79
85	263.27
90	278.76
95	294.25
100	309.74
105	325.22
110	340.71
115	356.2
120	371.68
125	387.17
130	402.66
135	418.14
140	433.63
145	449.12
150	464.6
155	480.09
160	495.58
165	511.06
170	526.55
175	542.04
180	557.52
185	573.01
190	588.5
195	603.98
200	619.47
205	634.96
210	650.44
215	665.93
220	681.42
225	696.9
230	712.39
235	727.88
240	743.36
245	758.85
250	774.34
255	789.82

b	7	Robot Wheelbase
Cturn	21.99	Turn Circumference
Cwheel	7.1	Wheel Circumference

Angle to turn	# of Degrees
0	0
260	805.31
265	820.8
270	836.29
275	851.77
280	867.26
285	882.75
290	898.23
295	913.72
300	929.21
305	944.69
310	960.18
315	975.67
320	991.15
325	1006.64
330	1022.13
335	1037.61
340	1053.1
345	1068.59
350	1084.07
355	1099.56
360	1115.05