

DR

Visual Logic

for HOVIS Genie

- Introducing DR-Visual Logic
- Starting HoVis Genie Programming With DR-Visual Logic
- DR-Visual Logic User Interface
- DR-Visual Logic Example Basic Program for Each Module



CONTENTS



01. Introducing DR-Visual Logic

1.1 Installation

02. Starting Hovis Genie Programming With DR-Visual Logic

2.1 First Program Step by Step

2.2 Program Compile, Download, and Run

03. DR-Visual Logic User Interface

3.1 Programming Module

3.2 Programming Module › Regular Type Module

3.3 Using Regular Type Module

3.4 Programming Module › Flow Type Module

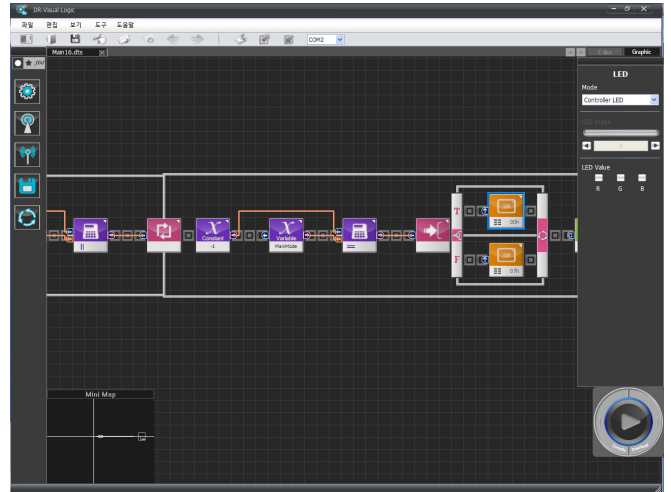
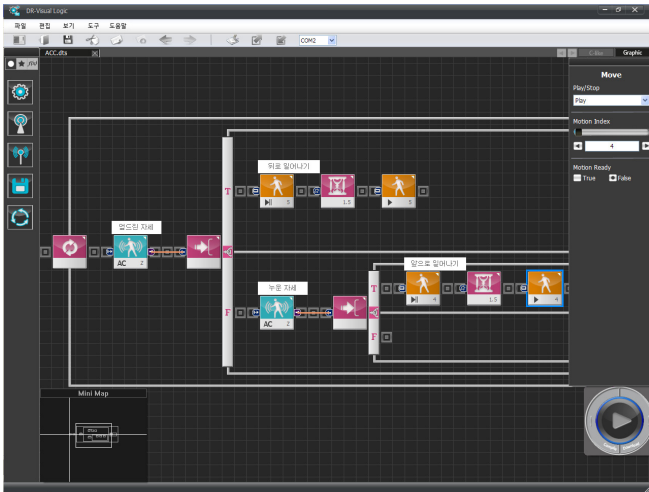
3.5 Programming Module › Pin

3.6 Programming Module › Connector Type

3.7 Property Window

3.8 Compile/Download

3.9 Diverse Action



04. DR-Visual Logic (HOVIS Genie)

4.1 Robot Motion

4.2 Individual Control of Upper Body Motors

4.3 Head LED Control

4.4 Forward Obstacle Detection

4.5 Touch Sensor and Sound Output

4.6 Navigation Movement.

4.7 Remote Control Drive.

4.8 Vision Application.

01

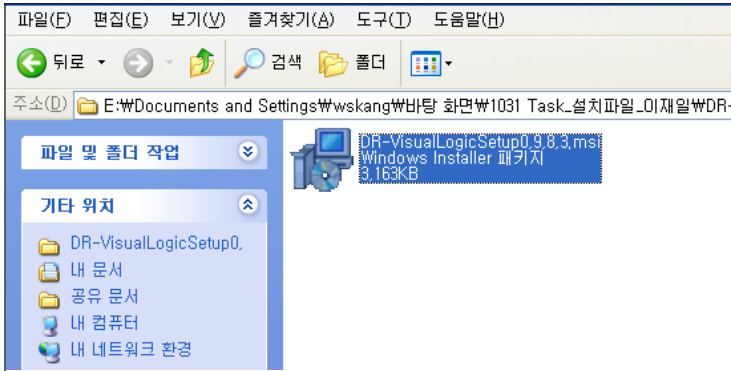
Introducing DR-VisualLogic

DR-Visual Logic is a modular graphic robot programming tool derived from the proprietary robot programming language developed by Dongbu Robot. Drag & drop programming method used by the DR-Visual Logic enables even the novice users to use DR-Visual Logic with ease. DR-Visual Logic also provides C-like tab which allows text codes converted from the graphic program to be viewed immediately. Since DR-Visual Logic program codes are similar to the C language, using DR-Visual Logic will also assist the novice programmers in learning the C language in the future. DR-Visual Logic is one of the easiest and yet powerful programming tools in the market and its versatility makes it equally popular with novice and advanced users alike. Planned upgrades to the program such as upgraded DRC function modules, motion modules, and integrated simulation will make the program even more powerful and versatile.

■ System Requirement

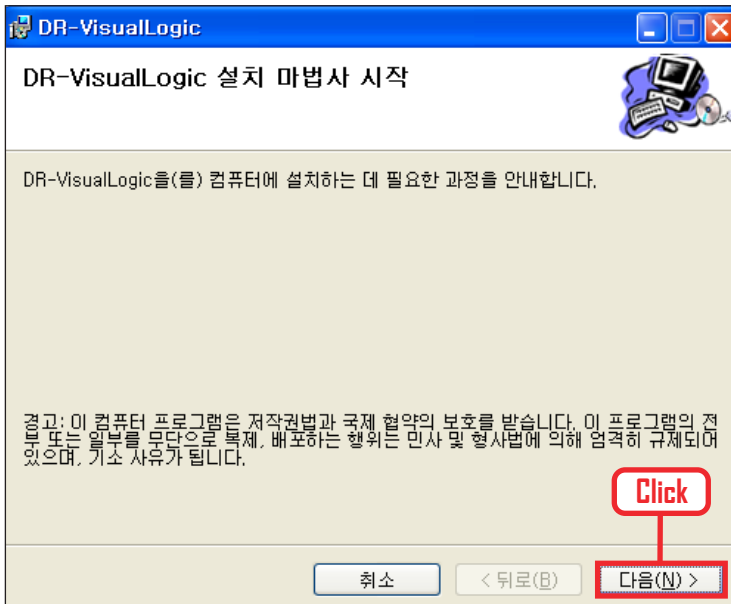
- Minimum Interl Pentium 800 Mhz
- Windows XP/Vista/7
- Minimum 256 MB RAM
- Minimum 20 MB hard disk space required
- USB Port
- Wi-Fi environment (Wireless AP required)

1.1 Installation / Step by Step from Installation to Execution



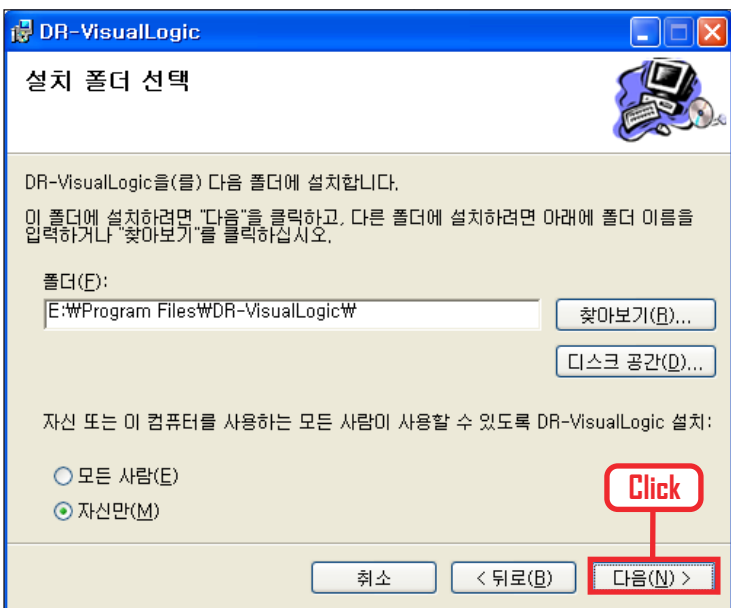
01 Installation file

Click Hoivis Genie DR-Viusal Logic installation file to start the installation.



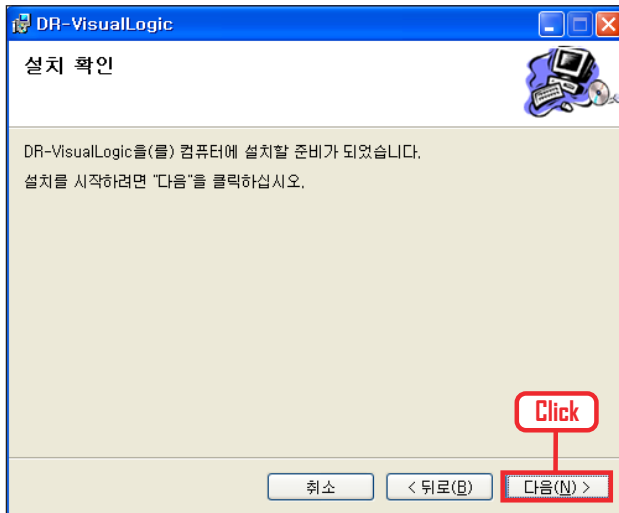
02 Start installation wizard

Click "Next" button.



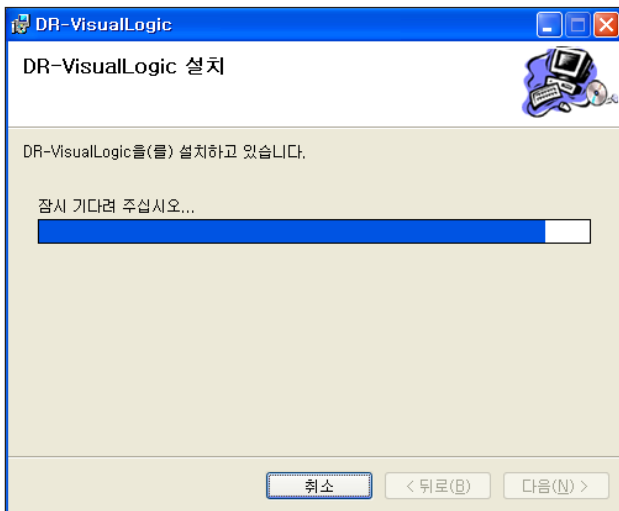
03 Select installation folder

Click "Next" button.



04 Confirm installation

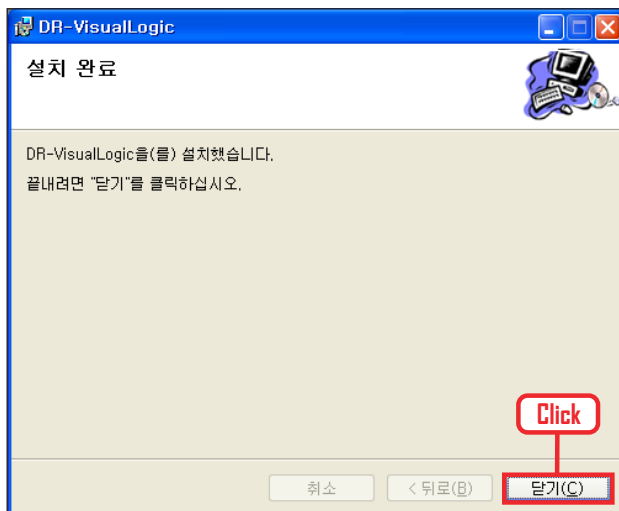
Click "Next" button.



05 Begin installation

Begin Installation.

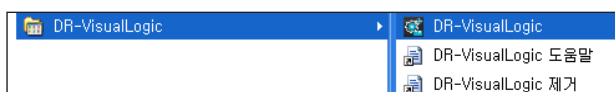
Wait until the progress bar ends.



06 Complete installation

Click "Close" button.

Software installation completed.



07 Check executable file

Check for the executable file, desktop shortcut icon and from Windows Start > All Programs > Dongbu Robot > DR-VisualLogic.

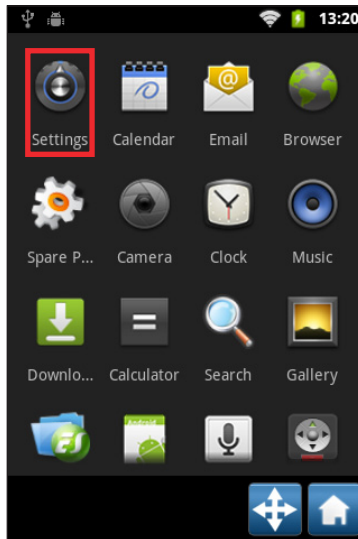
Click on the executable file to run the program.

02 Starting Hovis-Genie Programming with DR-Visual Logic

Wi-Fi Connection

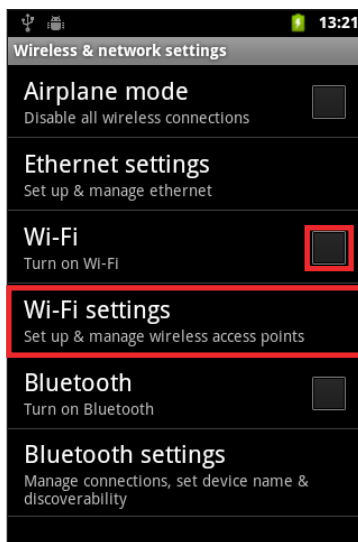
DR-Visual Logic and robot MID should be connected by Wi-Fi (wireless AP) prior to creating first DR-Visual Logic program.

Instructions for connecting MID to the wireless AP is as follows.



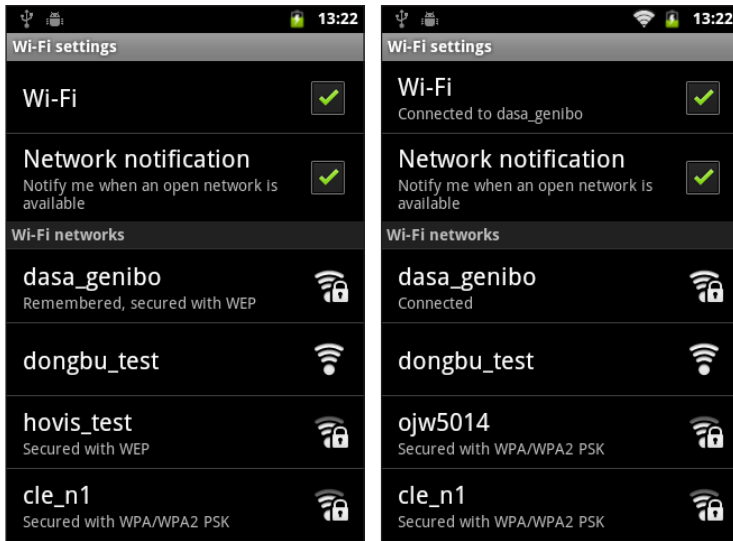
01 MID power ON

Turn on MID. Press the power button for approximately 3s to turn on the MID power. Press the menu button at bottom right corner after MID completes booting.



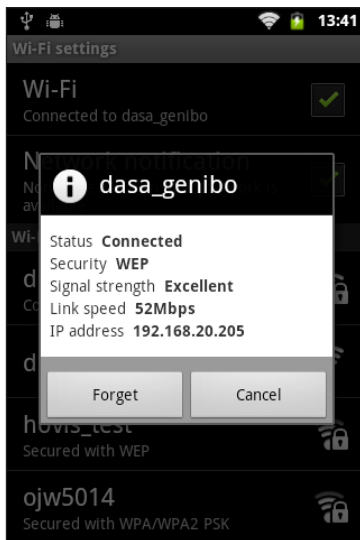
02 Select Wireless and Network

Turn on Wi-Fi by clicking "Wi-Fi" from Settings > Wireless and Network and then click W-Fi setup.



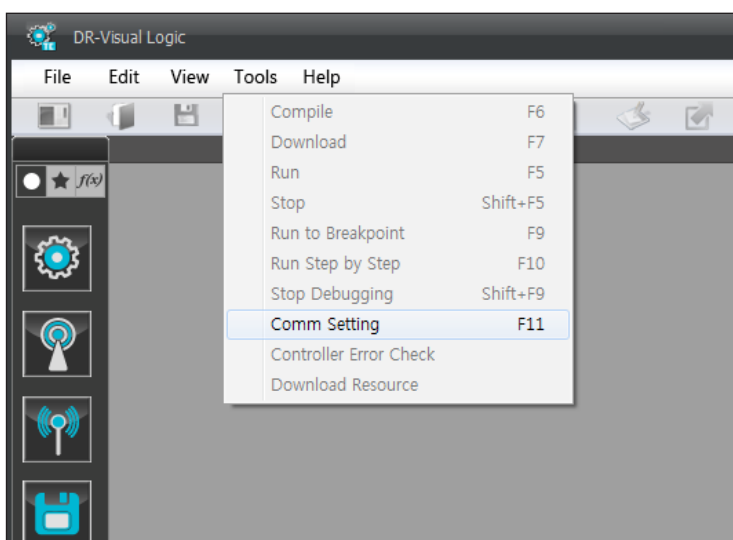
03 Connect MID to the AP PC is connected to.

Connect PC to the wireless AP. From MID, select the same AP PC is connected to. (In this example, PC is connected to AP dasa_genibo) Connection is successful when dasa_genibo shows "Connected" and Wi-Fi shows up at top of the window (right diagram).



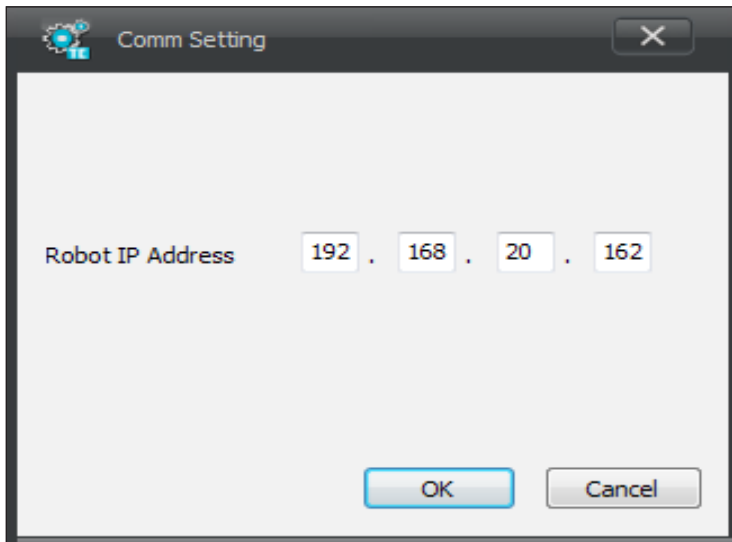
04 Check IP

Select wireless AP dasa_genibo again and check the IP .
(IP address 192.168.20.162)



05 DR-Visual Logic communication setup

From DR-Visual Logic, select Tool > Communication setup (Shortcut Key F11).



06 DR-Visual Logic communication setup

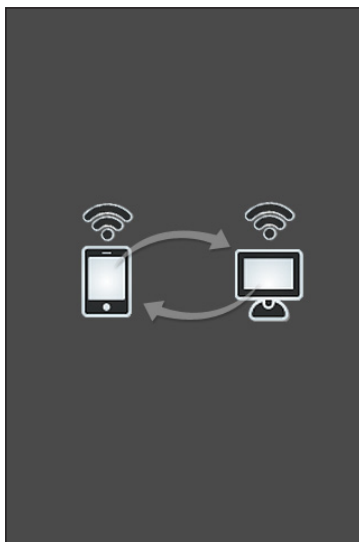
Enter the IP address from step 04 and click “OK” button.



07 Connecting DR-Visual Logic and MID

Click on the “Connect” button found on the top right corner of the screen.

MID screen will change as shown in the left with an announcement stating successful connection to DR-Visual Logic. This completes the Wi-Fi connection between the PC and MID.



- If connection does not occur,
 1. Check to make sure PC is connected to the correct AP
 2. Run DR-Visual Logic Player App from MID one more time.

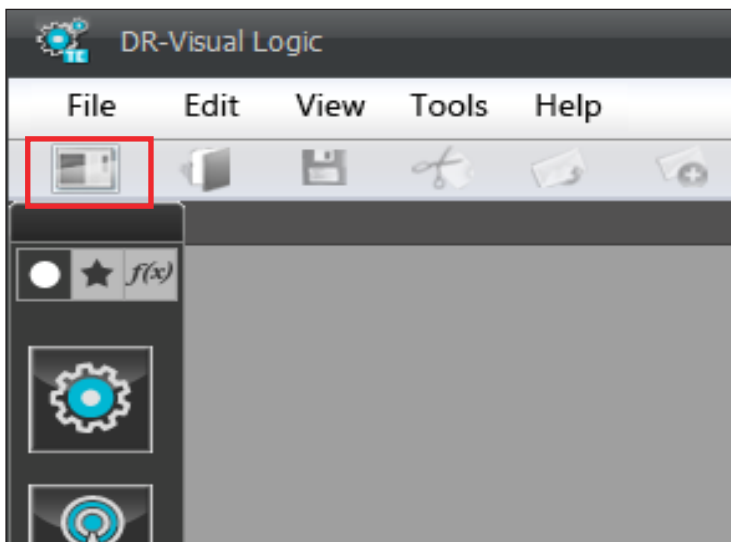
2.1 First Program Step by Step

[Example Description]

First example program. Robot will say “ Hello, I am Hovis Genie” and turn to the left and to the right when this program is run.

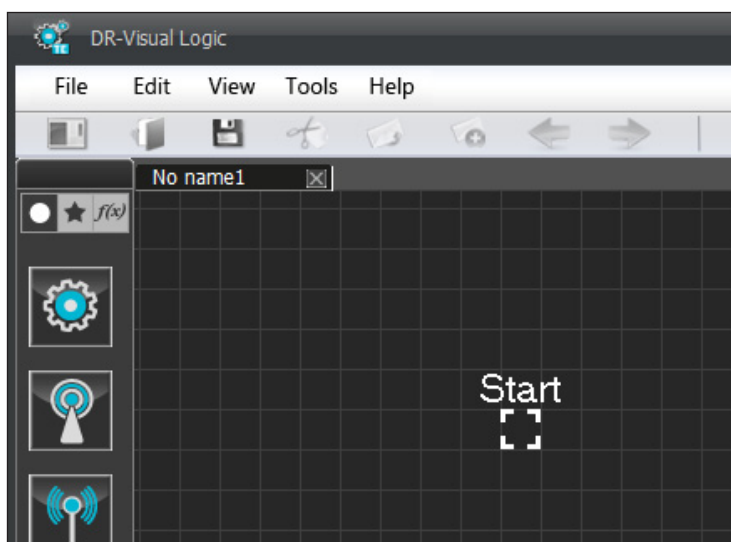
Tip. DR-Visual Logic screen control

- Mouse wheel up (Zoom in)
- Mouse wheel down (Zoom out)
- Shift + L mouse click and move (screen movement)



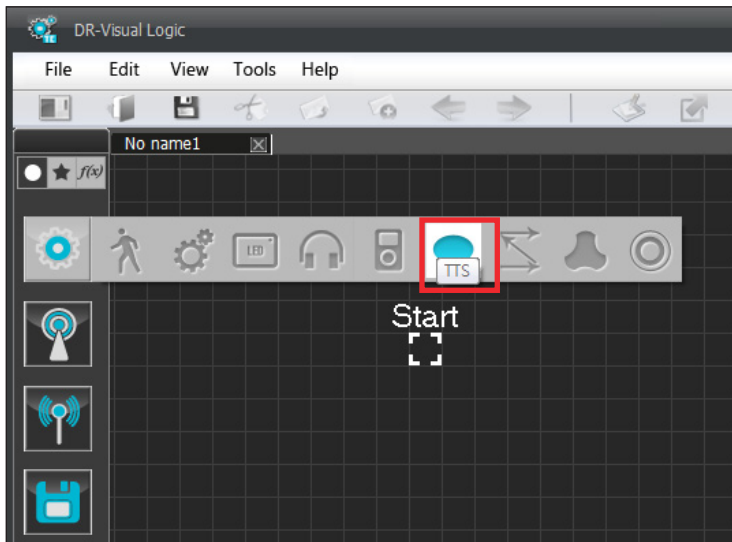
01 Create New

Create new dts file by clicking on the leftmost icon from the tool bar (Shortcut Ctrl+N).



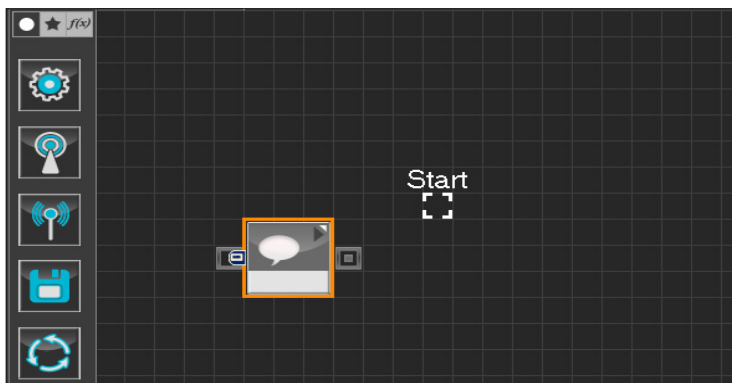
02 New Window

New file created.



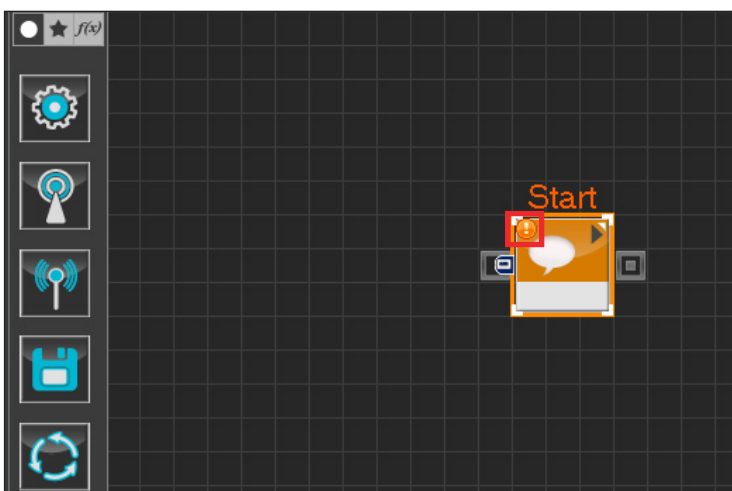
03 Module selection

Click on the module from the left module bar prior to placing the module. Click Motion > TTS module.



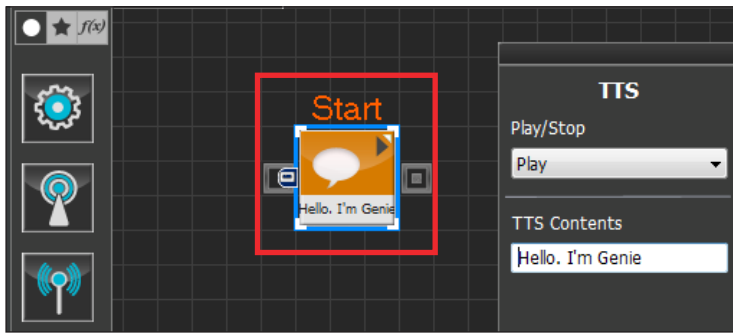
04 Placing module

Click on the module and module will move following the mouse cursor. Click the mouse button once again to place the module at current location. Drag the module to the Start Point.



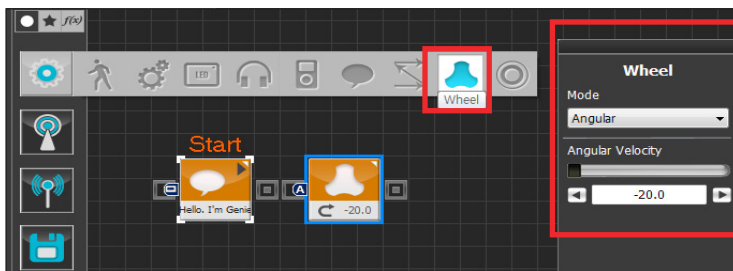
05 Start Programming

When the module and the Start Point is docked properly, module will become active and change color as seen in the diagram to the left. This means programming has started.



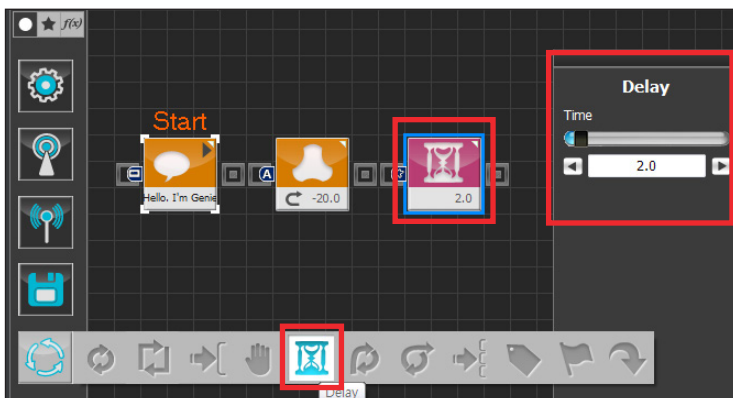
06 Resolving warning

Small exclamation mark will appear at top left corner of the module when unresolved warning exists. Place the cursor on top of the exclamation mark to see the details. In this particular case, warning appears due to missing TTS content. Enter the following TTS content, " Hello, I am Hoavis Genie."



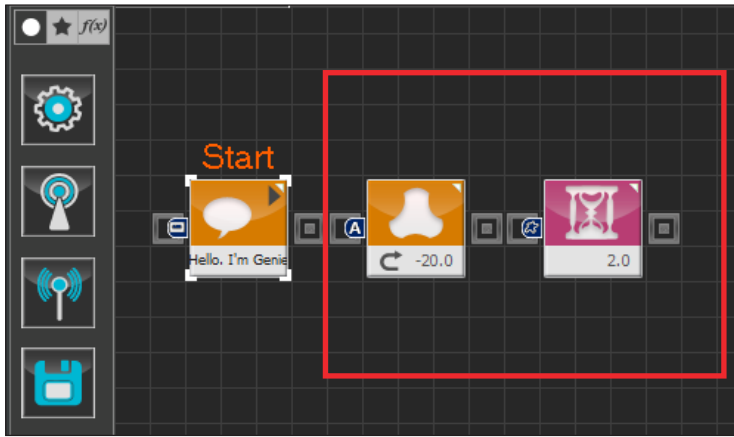
07 Wheel Control

Select Motion > Wheel module and place the module as shown in the diagram to activate the module. Select the placed Wheel module and set Mode to Angular and Angular Velocity to -20. Angular Mode is selected to rotate the robot, Robot rotates to the right when Angular velocity is - and to the left when value is +.



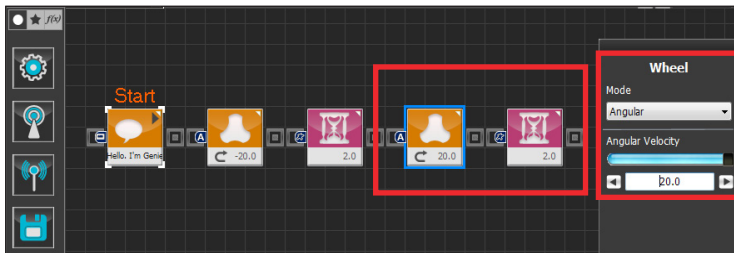
08 Delay

Delay 2s before executing next command. Current wheel command to rotate to the right will be maintained for 2s.



09 Module Copy

Next, left rotation will be maintained for 2s. Copy and paste two modules created in steps 07–08 and change the Wheel module Angular Velocity. Drag and select two modules as shown in the diagram and press Ctrl+C.



10 Paste Module

Press Ctrl+V to paste the copied module from step 09. Change the copied Wheel module Angular Velocity to +20.



11 Stop Wheel

Set Wheel module Angular Velocity to 0 to stop the Wheel.

```

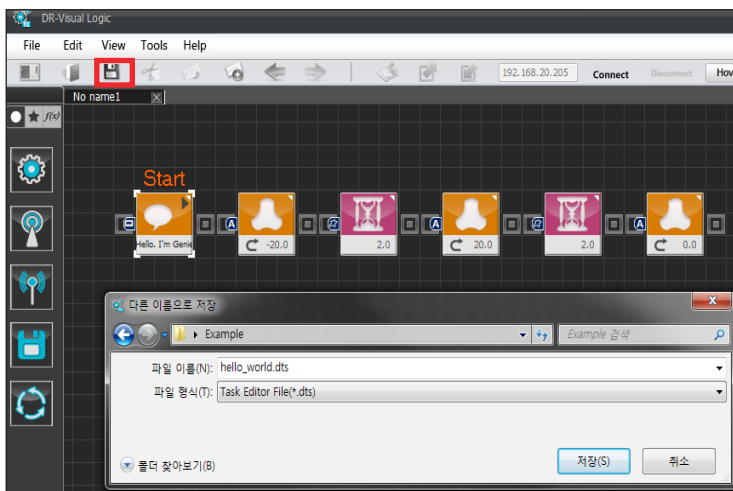
1 void main()
2 {
3     mid_tts_play( "Hello. I'm Genie" )
4     mid_wheel_angular( -20.0 )
5     delay( 2000 )
6     mid_wheel_angular( 20.0 )
7     delay( 2000 )
8     mid_wheel_angular( 0.0 )
9 }

```



12 View C-like

Click the 'C-like' tab near the top right to open the window as shown in the diagram to the left. DR-Visual Logic codes are very similar to the C language structure so studying the codes will help the user become familiar with the C language. Cursor in the window will jump to the corresponding txt line each time different module is clicked. This enable the user to easily view the codes related to each module.

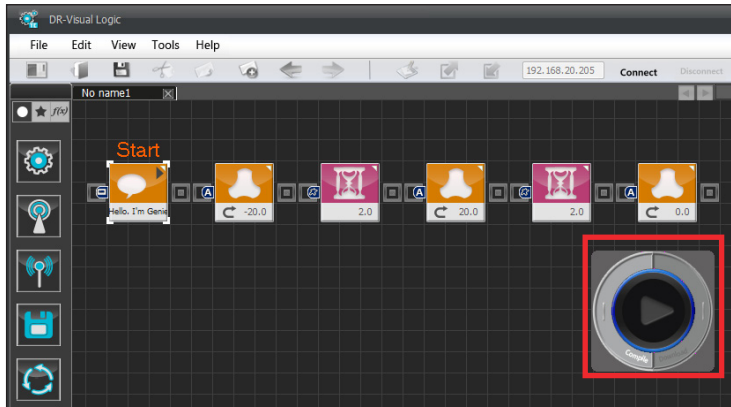


13 File Save

Click on the third icon in the tool bar to save the file (Shortcut Ctrl+S).
(hello_world.tts)

2.2 Program Compile, Download, and Run

In previous sections, we have connected PC(DR-Visual Logic) to the MID through the wireless AP and created first example program. In this section, program will be compiled, downloaded to MID and run.



01 Compile, download, and run

After programming is complete, created program will be compiled → downloaded → and run.

Use the remote control window to compile, download, and run the program.



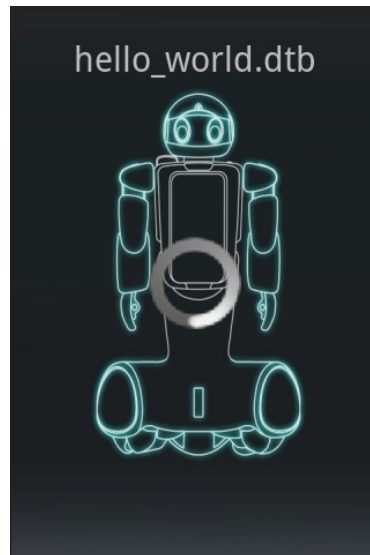
02 Compile

Click Compile to compile the program. If error occurs, check the error message and take appropriate action. (Compile converts dts file to executable dtb file)



03 Download

Click Download to download compiled file to the MID. MID will change to the screen shown on the left and download will start automatically as long as there is no problem with the connection between the PC and the MID. (Step 02 can be skipped if desired since file will be compiled automatically when Download is clicked).



04 Run

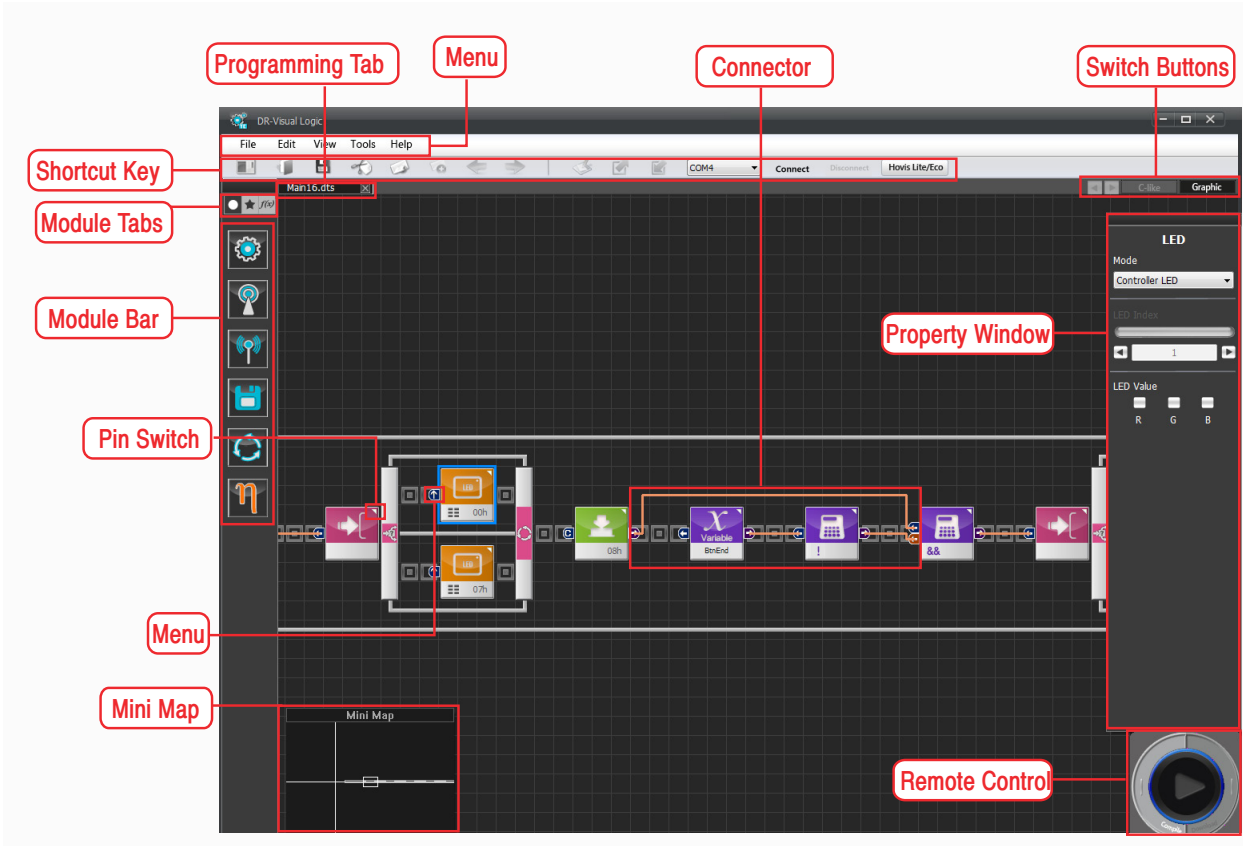
Run the downloaded program by clicking on the middle Run button.

MID will change to the screen shown on the left and program will run.

※Caution

- Make sure to download prior to clicking the Run button.
- Check the robot power if program does not run.

03 DR-Visual Logic User Interface

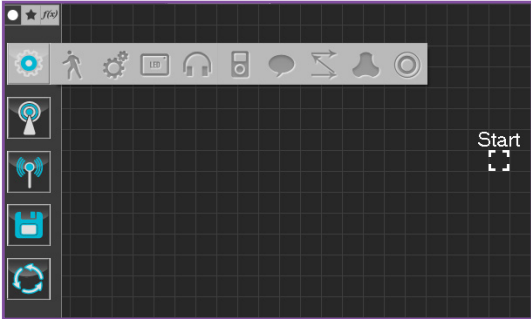
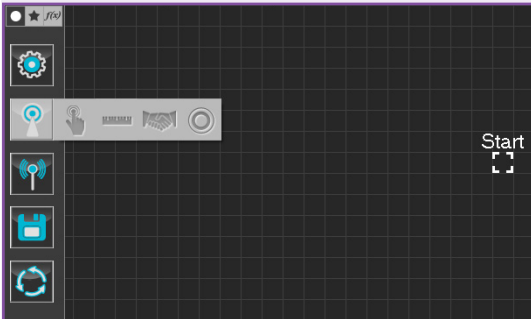


- 1 Menu : Composed of DR-Visual Logic function menus File/Edit/View/Tools/Help
- 2 Tool Bar : Collection of frequently used function icons
- 3 Module Tabs : Tabs for selecting module bar. There are 3 different types of module bars; All/Favorite/My Module.
- 4 Module Bar : Bar found on the left side of the screen for selecting and adding modules. All bar is composed of 5 types of module packs ; Motion / Sensor / Communication / Data / Flow. Hovis Lite/Eco has an extra Third Party Pack (ETA) in addition to the 5 packs mentioned.
- 5 Programming Tab : Enables the user to view the name of the file currently being edited and switch between windows if multiple files are being edited at the same time.
- 6 Switch Buttons : Composed of L/R buttons enabling the user to shift L/R to view the hidden programming tabs when multiple files are open simultaneously with number of programming tabs going beyond the window width. Buttons for switching between the graphic programming window and C-like txt window.
- 7 Property Window : Window for entering module properties. Modules have various properties which can be set using the property window. Some property values maybe entered using the input pin.
- 8 Remote Control : Provides convenient access to frequently used commands Compile/Download/Run.
- 9 Mini Map : Shows the overall picture of the program and current location. Quickly jump to another location in the program by clicking on the desired program location in the minimap.
- 10 Pin : Divided between input and output pins. Used when output value of one module is used as input value of another module.
- 11 Pin Switch : Toggles on/off the pin name.
- 12 Connector : Line connecting the pins.

3.1 Programming Module

DR–Visual Logic is composed of following modules.

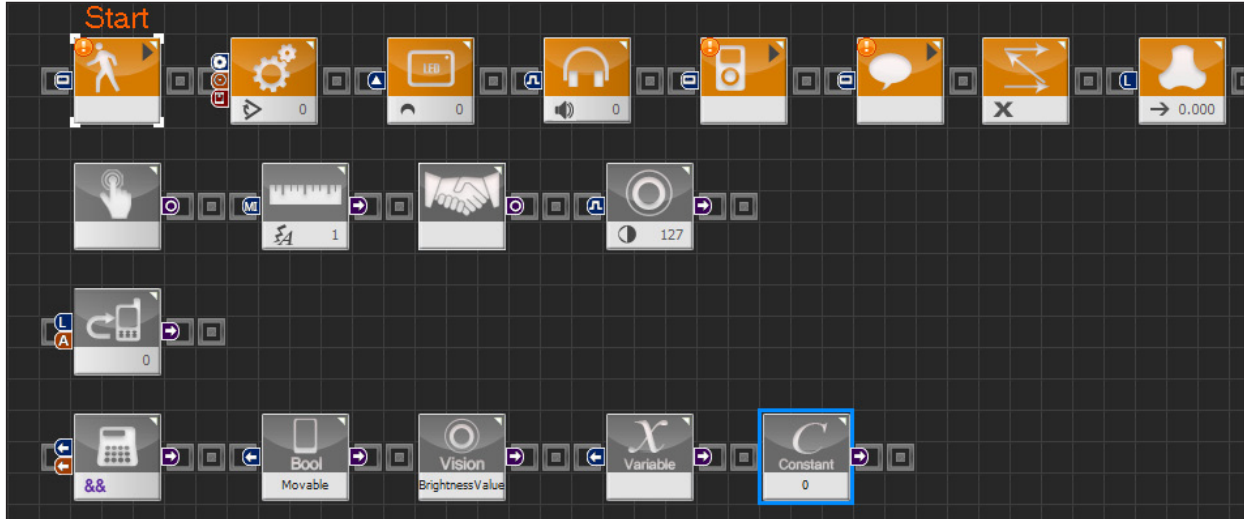
Module Pack contains all programming modules required to create a program. Modules in DR–Visual Logic for HOVIS Genie are composed of functions supported by the MID.

Module Pack	Picture	Module	Description
Motion		Move	Run saved robot motion
		Motor	Individual motor speed/position control
		LED	Head LED, eyes, ears, nose, mouth, forehead LED control
		Sound	Run assigned melody
		MP3	Play assigned MP3 file
		TTS	Convnet Text to speech.
		Navigation	Set standard of reference in 2d plane and add x, y coordinates to move the robot
		Wheel	Set Linear and Angular speed
		Vision Capture	Capature video from MID front camera
Sensor		Touch Sensor	Head module touch detection
		Distance Sensor	Measure distance 5 at front 3 at lower section
		Hand Touch Sensor	Detect palm tact switch press
		Vision Sensor	Video process captured screen

Module Pack	Picture	Module	Description
Communication		IR Receive	Recognize IR remote control data
Data		Operator	Operator(logical/arithmetic/relational/bitwise/incremental/decremental)
		MID RAM	Used to read or use assigned variable used by MID
		Vision Result	Used for reading the video process value of caputred screen
		Variable	Used for using user assigned vaiable
		Constant	Used fo reading constant value
Flow		Loop	Infinite loop/for statement
		While	while statement (repeat if condition is true)
		If-else	if-else statement (Control branch)
		Wait	Wait during specific condition
		Delay	Wait for specified duration
		Continue	Go back to the beginning of the loop
		Break	Exit loop or switch-case statement
		Switch	Switch in switch-case statement
		Case	Case in switch-case statement
		Label	Assign specific location of the program with label
Goto	move to assigned label		

3.2 Programming Module › Regular Module

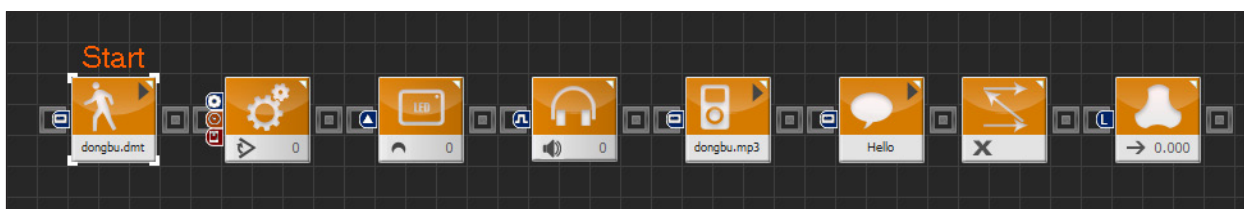
Regular modules are used by connecting the modules to each other in order. All modules are Regular modules except for the Flow modules.



From the top, module icons represent Motion, Sensor, Communication, and Data modules.

3.3 Using Regular Modules

Module in motion module pack does not have output and can generate source codes even by itself.

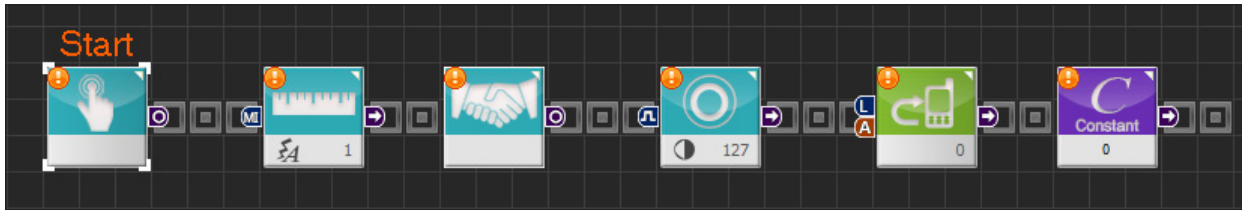


```

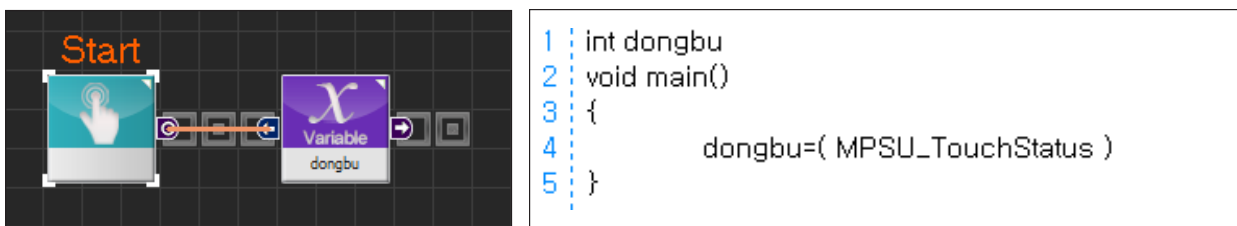
1 void main()
2 {
3     mid_motion( "dongbu.dmt", 0, 0 )
4     jog( 512, 0, 0, 60 )
5     mid_led_brow( 0 )
6     mid_sound( 0 )
7     mid_mp3_play( "dongbu.mp3" )
8     mid_ts_play( "Hello" )
9     mid_navi_init()
10    mid_wheel_linear( 0.000, 0, false )
11 }

```

Sensor module, Communication module, and Constant in Data module cannot generate source codes by themselves. These type of modules have meaning only when their output pin is connected to an input pin of another module. Error is shown on top left corner of the module when source codes cannot be generated.



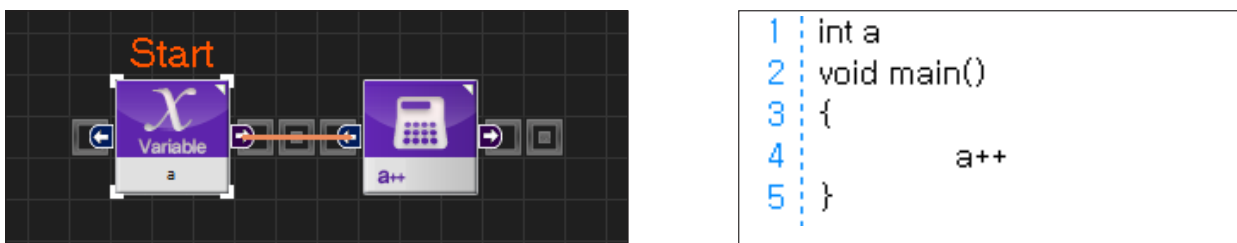
Error will disappear from the module when output pin becomes connected to an input pin of another module and C-like source codes are generated.



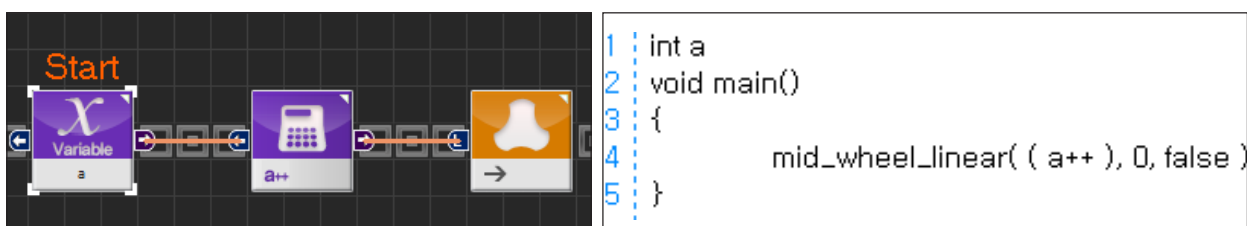
Operator modules will generate source codes only when output pins are connected to input pins of their modules.



There is one exception to the above. Incremental/Decremental operator can become a source for increasing the value of the variable even without the outp pin connection if it is connected to a variable whose value can be changed.



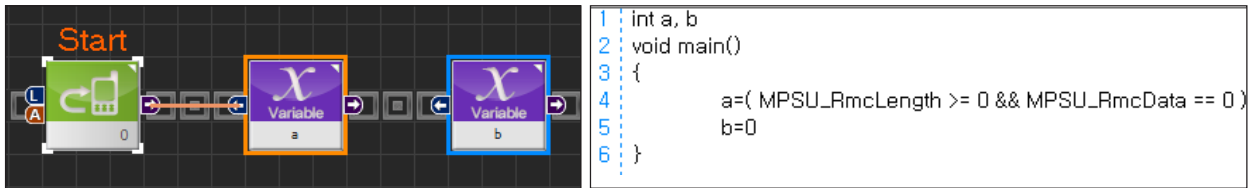
When output pin becomes connected to another module, source that increases the value of the variable becomes incorporated into the source of the connected module.



Other modules such as MID RAM, Vision Result, Variable do not generate source codes by themselves.



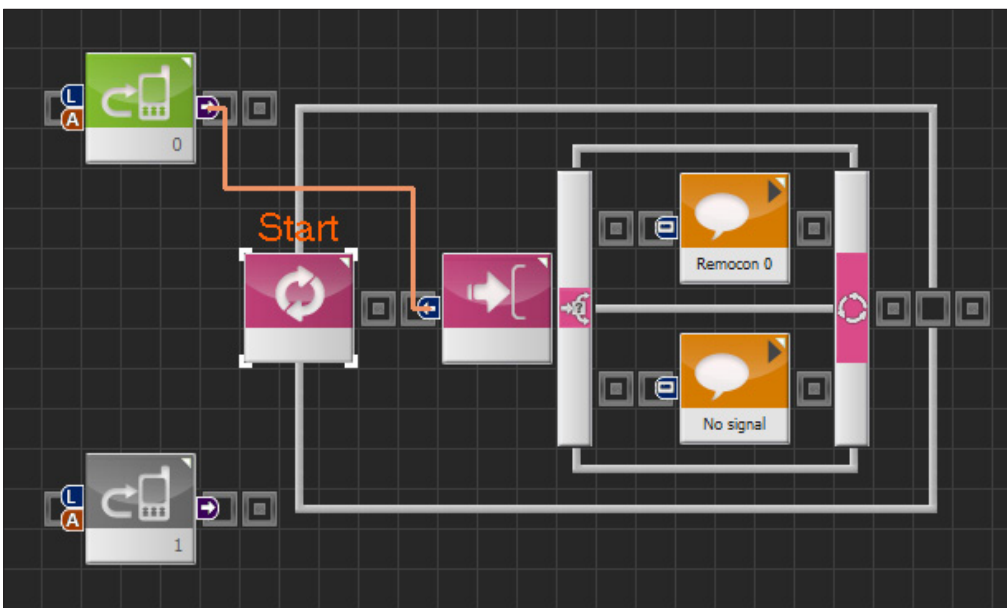
However, when other module is connected to the input pin, module becomes a source of value for the variable even without connection to the output pin. Also, when Assign Constant Value is checked True in the property window, it becomes a source of value that will substitute the constatatnt value below.



Read-Only exists among the MID RAM and Vision Result. In this case, input pin disappears and substitution becomes impossible.

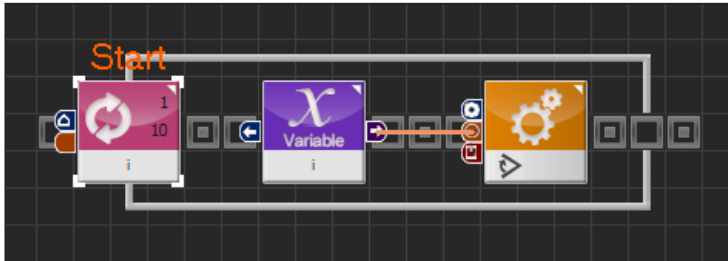


Modules that can generate source codes even without connection to the output pin must be on the program line that begins from Start in order to be activated and generate source codes. Modules that generate source codes through output pin can be outside of the program line and still generate source codes as long as connector is connected.



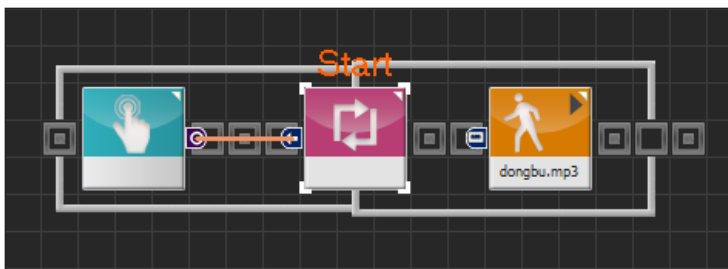
3.4 Programming Module › Flow Type Module

Flow modules connect to the regular modules and control the flow of the program with loop, switch, and etc. Unlike regular modules, outline appers around the flow modules when they are connected to the regular modules.



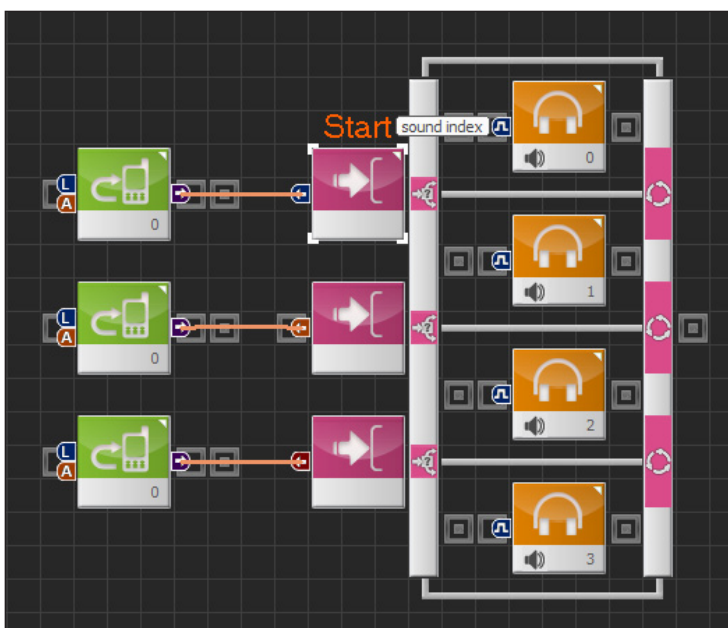
Loop

Loop module is used to repeat specific section of the program. Loop with For statement would repeat specific number of times whereas Loop with Forever statement would repeat infinitely.



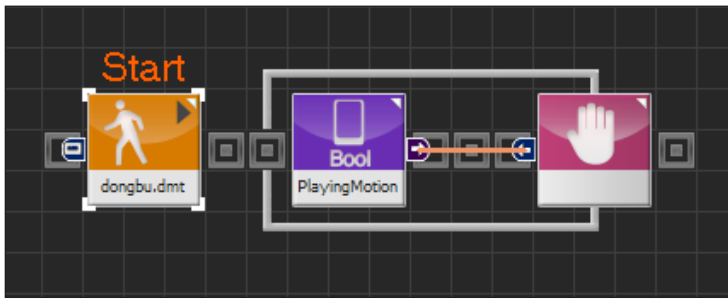
While

While module requires certain condition to be met before proceeding to the next step. It is a loop statement with attached condition.



If-else

If-else statement branches the program depending on the condition. Number of conditional statements can be increased/decreased by pressing the button in the property window.



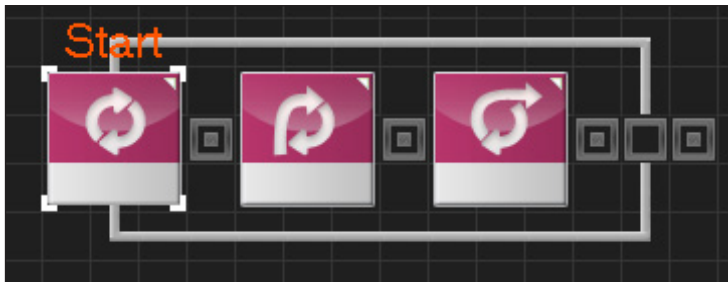
Wait

Stop program execution while input condition is true and start when when condition becomes false.



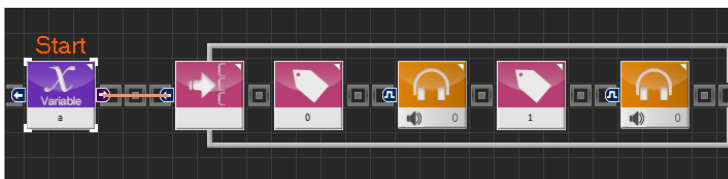
Delay

Delay program execution for specific period of time,



Continue, Break

Continue can only be used within the loop statement and sends the execution of the program back to the beginning of the loop. Break can only be used within the loop statement and switch-case statement, Break is used to exit from the loop and switch-case statements.



Switch, Case

These two modules are used together to form switch-case statement. Depending on the switch module input, task after the matching case statement will be executed.

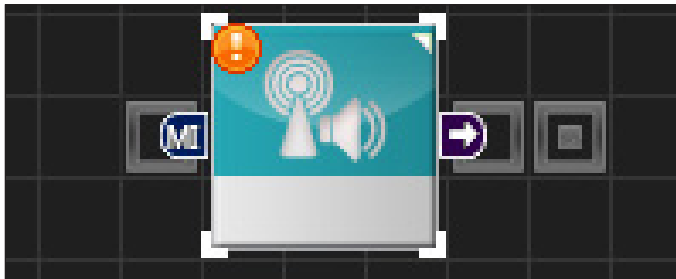


Label, Goto

These two modules are used together to perform jump function. Program will jump the the label module when goto statement with identical name as the label module is encountered.

3.5 Programming Module › Pin

Some modules have input and output values. Resulting value of an output pin becomes an input value of another connected module.



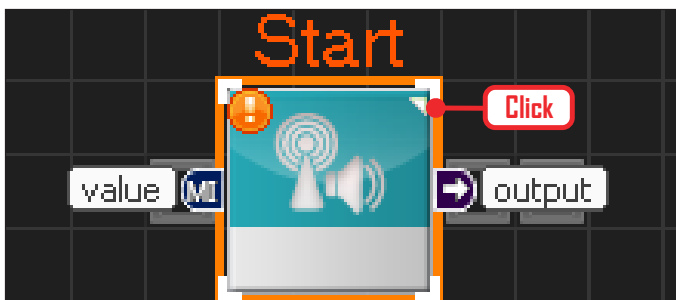
Pin

Modules with Input/Output values will have pins on left and right side of the module. Left side pin is the input pin and the right side pin is the output pin.



Help Ballon

It is difficult to distinguish the pin just by looking at the icon. To find out the function of the pin, place the mouse cursor on top of the pin and balloon will appear with the name of the pin.



Opening Help Ballons

Click on the triangle at top right corner of the module to see the name of several pins all at once. Pin names will appear beside each pin. Click one more time to close the balloons.

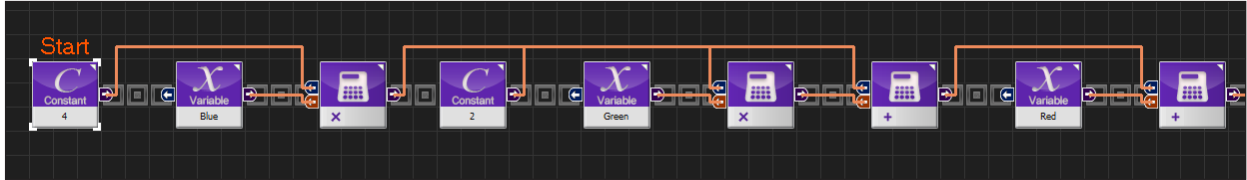


Pin Connection

To enter the output value of the previous module as input value of the following module, use the mouse to drag and connect. Connecting line will appear as shown in the left photo.

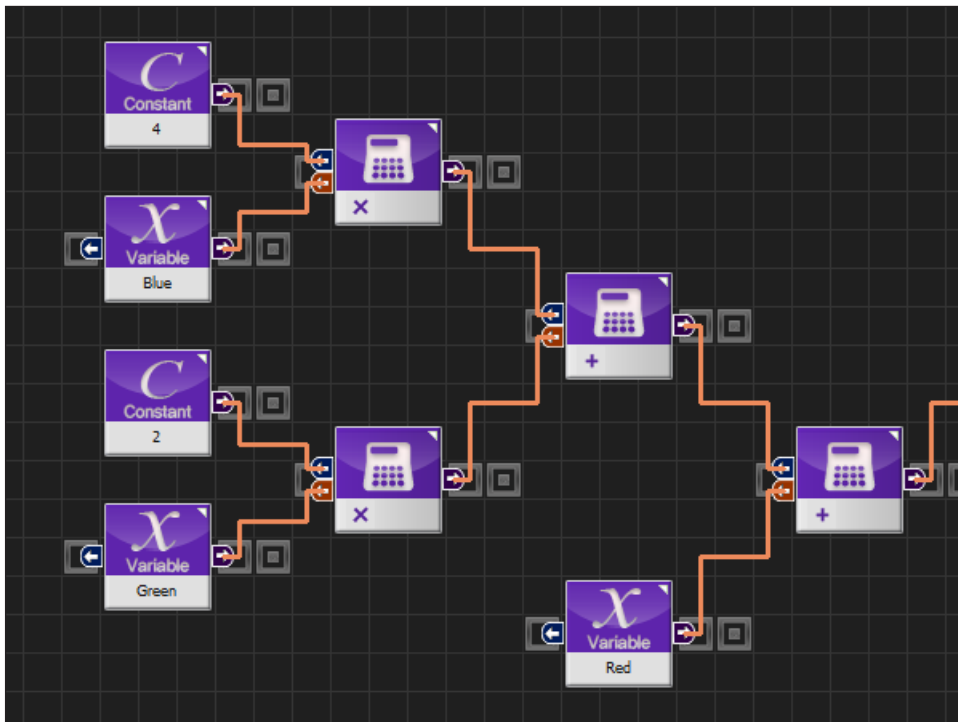
3.6 Programming Module > Connection Type

Modules can be connected by either serial or parallel method.



Serial Connection

In serial type connection, modules are connected sequentially from left to right. Photo above shows arithmetic operation program. $((4 \times \text{Blue}) + (2 \times \text{Green})) + (1 \times \text{Red})$ operation shown as serial type connection

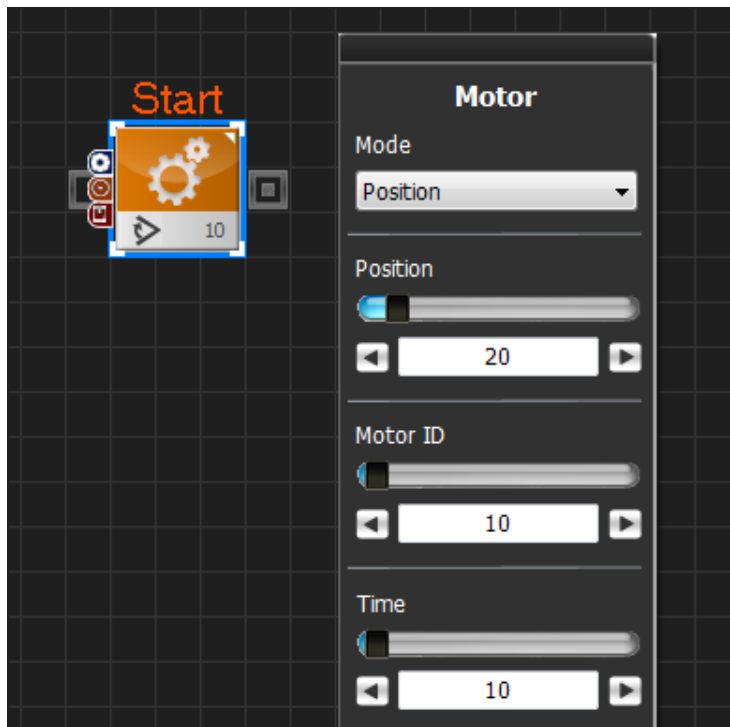


Parallel Connection

Parallel connection uses vertical space to connect the modules in parallel format. Bottom diagram showing parallel connection and the top diagram showing serial connection is an identical program.

3.7 Property Window

Modules have their own properties and these properties must be given a value for program to work. UI in property window includes list popup, radio button, number setting, and etc. Refer to the Help file for details on properties for each module, property values, and limits.

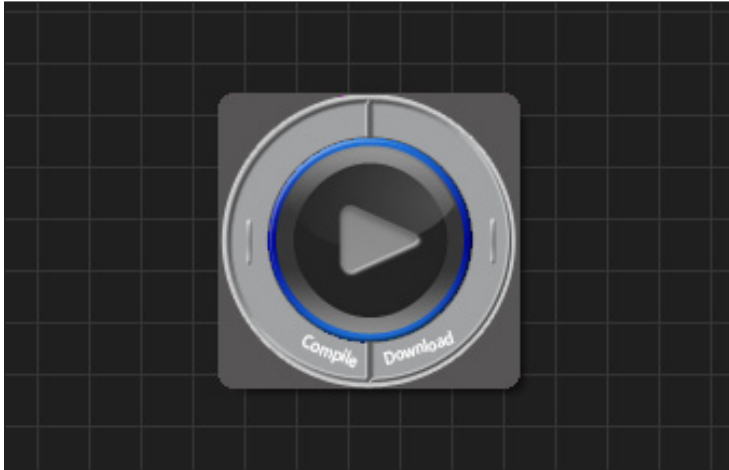


Property Window

When motor module is clicked, property window opens up on right side of the window. Motor has speed and position control properties. To control the position, select 'Position' in Mode selector. To control speed, select 'Velocity'. Position, Motor ID, Time values are adjusted in the detailed settings below the Mode Selector.

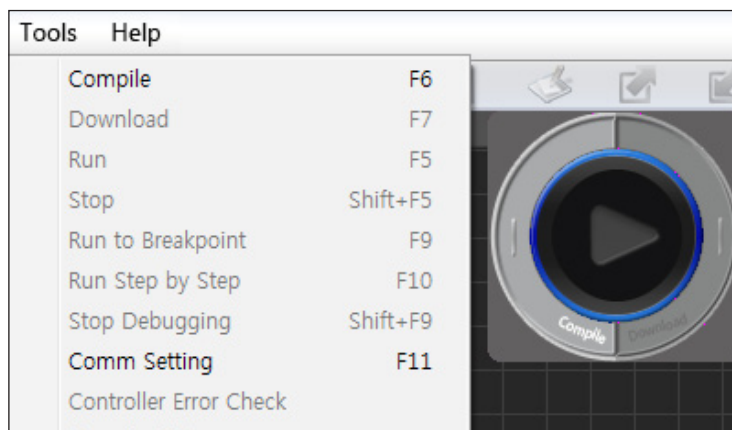
3.8 Compile/Download

Completed program is compiled, downloaded to the robot and run. Downloader is a large icon located at bottom left side of the programming window. More specific commands are found in the tools menu.



Downloader Icon

Downloader icon has three commands. Compile command on the left, download command on the right, and play command in the middle shown by arrow like icon.



Tools Menu

Commands can be run more precisely from the tools menu.

Compile : Compile program.

Download :
Download compiled program.

Run : Run downloaded program.

Stop : Stop the program.

Run to break point :
To be available in the future version.

Run step by step :
To be available the future version.

Resource download :
Download resources required by the program. (DMT, MP3)

3.9 Functions

Menu Functions

DR–Visual Logic menu functions are shown below in the function table.

Type	e	Description
File	New Window	Create new dts.
	Load	Load specified dts file.
	Close	Close open dts file.
	Save	Save current dts file.
	Save As	Save current dts file under another name.
	Print Preview	Print preview.
	Print	Print.
	Quit	End program.
Edit	Undo	Undo previous command.
	Redo	Redo previous command.
	Cut	Cut selected section to the clipboard.
	Copy	Copy selected section to the clipboard.
	Paste	Paste content of the clipboard.
	Delete	Delete selected section.
View	Default View	The button will show you the default scale screen.
	Zoom In	Enlarge Screen
	Zoom Out	Reduce Screen
	Center This Module	The button will relocate the selected module to the center of the screen. If you do not select any module, Starting point will be located in the center of the screen. When you click a line in the C–Like window, you can also see which module correspond to the clicked line in Graphic view.

Type	Name	Description
Tools	Compile	Compile program.
	Download	Download compiled program.
	Run	Run downloaded program.
	Stop	Stop program.
	Run to Breakpoint	Program will pause at the breakpoint designated in the C-like window. (To be available in future version)
	Run Step by Step	Program in the C-like window will be executed line by line. (To be available in the future version)
	Stop Debugging	Stop debugging. (To be available in the future version)
	Communication Setting	Wi-Fi communication setup.
	Controller Error Check	MID error check. (To be available in the future version)
Help	Index	open help file
	Online Help	Displays Donbu Robot website.
	S/W Update	Checking Software version
	F/W Update	Update controller's firmware (do not apply to Hovis Genie)
	F/W Restore	Restore firmware in case of failure of controller and detect incorrect firmware. (Do not apply to Hovis Genie)
	About	Displays window of program properties.

Other Functions

Click wheel button and drag: Moves screen.

Shift + left click and drag: Moves screen.

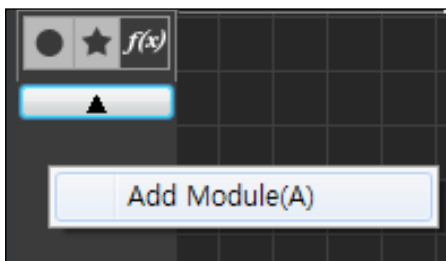
Ctrl + left click and drag selected module: Copy selected module.

Wheel Up : Zoom in

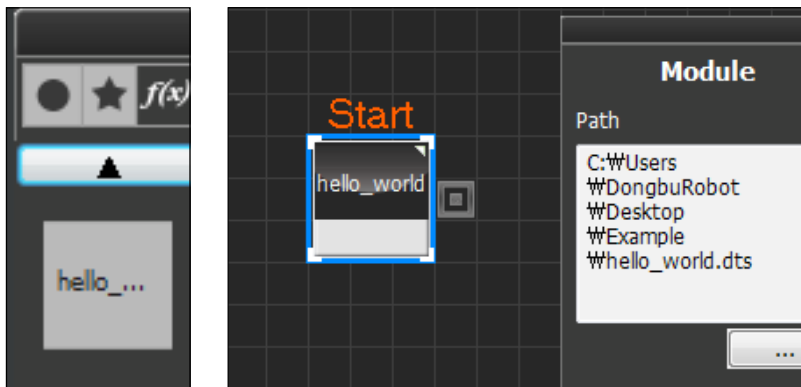
Wheel Down : Zoom out

Using My Module

dts file created with DR-Visual Logic can be called from another dts file and used like a function.



Click My Module from the module tab and then right click to open the file selection window. Select dts file to add from the file selection window.



My Module has been added. Click and place the dts file to use the file as modularized function.

```

1 void hello_world()
2 {
3     mid_tts_play( "Hello. I'm Genie" )
4     mid_wheel_angular( -20.0 )
5     delay( 2000 )
6     mid_wheel_angular( 20.0 )
7     delay( 2000 )
8     mid_wheel_angular( 0.0 )
9 }
10 void main()
11 {
12     hello_world()
13 }

```

Switch to C-like window to view the source codes and notice how added module is being used like a function.

04

DR-Visual Logic (HOVIS Genie)

Example Basic Program for Each Module

Example basic program for each type of module to control Hovis Genie using DR-Visual Logic.

Example Title	Applicable Module	Example Description
Robot Motion	Motion › Move	Program for repeating Taekwondo motion continuously. This particular program makes an excellent robot demonstration program. (Filename: exam_motion.dts, tekwuando.dmt)
Individual Control of Upper Body Motors	Motion › Motor	Program to control upper body motors individually. This program can be used instead of motion file when creating simple motion. (Filename: exam_motor.dts)
Head LED Control	Motion › LED	Eyes, ears, mouth, forehead LED control program. Use robot LEDs to create various special effects. (Filename: exam_led.dts)
Front Obstacle Detection	Sensor › Distance Sensor Motion › TTS	Program will move the robot forward and come to a stop and output voice when it detects an obstacle. Program can be used to create an autonomous movement program with obstacle avoidance. (Filename: exam_distance_tts.dts)
Touch Sensor and Sound Output	Sensor › Touch Sensor Sensor › Hand Touch Sensor Motion › Sound Motion › Mp3	Program will output MP3 and other sound effect output when it receives input through the touch sensor and tact switch sensors. Use robot sensors and multimedia to provide HRI(Human Robot Interaction). (Filename: exam_touch_sound.dts, song_001.mp3)
Navigation Movement	Sensor › Touch Sensor Sensor › Hand Touch Sensor Motion › Sound Motion › Mp3	Program will set navigation waypoints for robot to follow. (Filename: exam_navigation.dts)
Remote Control Drive	Communication › IR Receive Motion › Wheel	Program will use the remote control to drive the robot. (Filename: exam_remocon.dts)
Vision Application	Motion › Vision Capture Sensor › Vision Sensor	Color Tracking Vision application program using the camera installed in the MID. Example enables the robot to recognize and track red object. (Filename: exam_vision_color_tracking.dts) – Other application examples (Only files provided) Vision application program provides very powerful sensor function. Various application programs can be created. Linetracer (Filename: exam_vision_line_tracer.dts) Motion Detection Game (Filename: exam_vision_motion_game.dts)

4.1 Robot Motion

Example description

Robot motion is created by controlling the motion of each individual motor which could become very complex and time consuming. To simplify creating robot motion, robot motion editors are used in most cases to create robot motion.

In this example, motion file (DMT) created using motion editor DR-SIM will be downloaded to the robot and run continuously. Example will use Taekwondo motion tekwuando.dmt which can be found at our website together with the example.

Entire Program

1. Loop – Infinite loop
2. Move – Run Taekwondo motion
3. MID RAM (PlayingMotion) – Output True if motion is running and False otherwise.
4. Wait –Repeat applicable loop if input pin value is TRUE.

Graphic



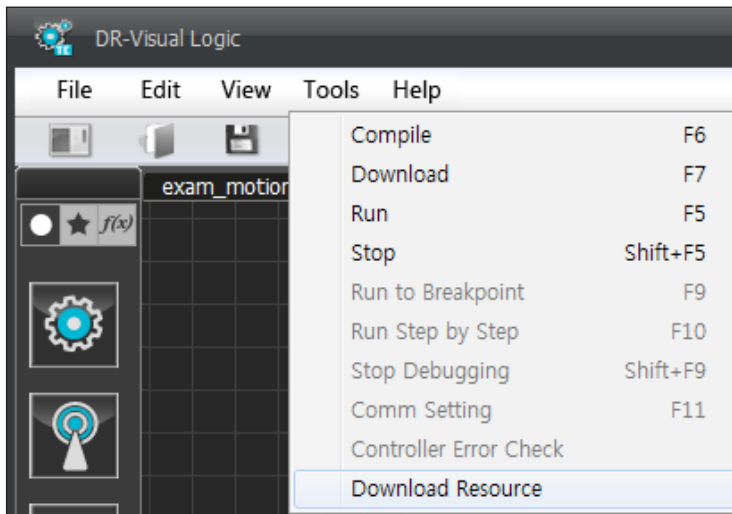
C-Like

```

1 void main()
2 {
3     while( true )
4     {
5         mid_motion( "tekwuando.dmt", 0, 0 )
6         waitwhile( MID_PlayingMotion )
7     }
8 }

```

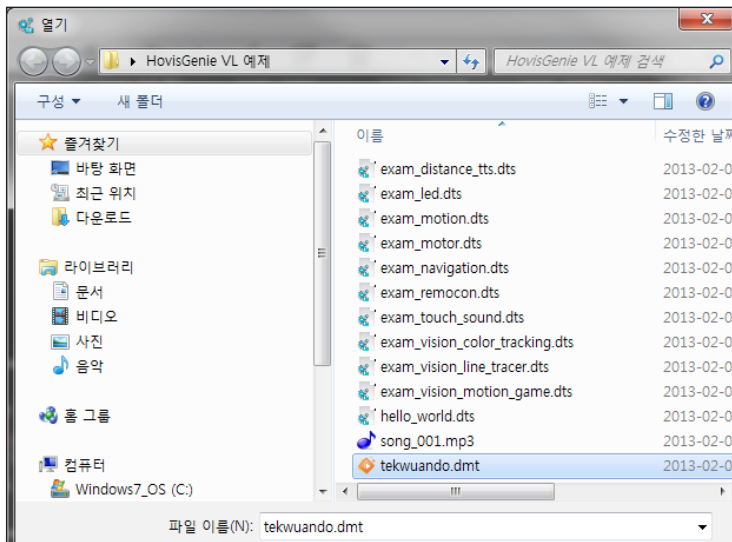
Step by Step

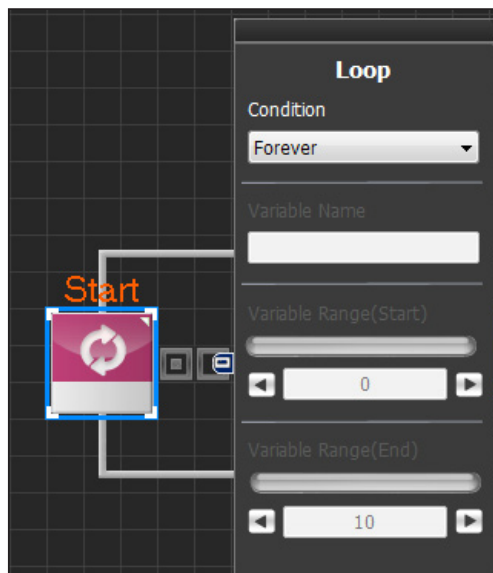


01 Preparation

Connect DR-Visual Logic to MID and download Taekwondo motion file.

- Menu > Tool > Click resource download
- Select Taekwondo motion file and start download.





02 Loop

Select Loop module and dock to the Start Point.

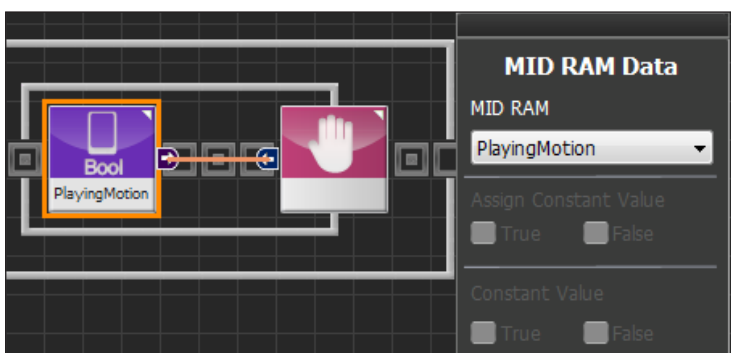
Since Taekwondo motion file will be repeated continuously, set the Condition in the property window to Forever.



03 Move

Place Move module as follows.

From the property window, set Play/ Stop to Play and select the motion file name to play.



04 Wait and MID RAM Data

Place Wait module in succession. Place MID RAM Data in the Wait module and select PlayingMotion in the property window.

PlayingMotion will output True if motion is being played and output False otherwise. This output value will be connected to the input pin of the Wait module.



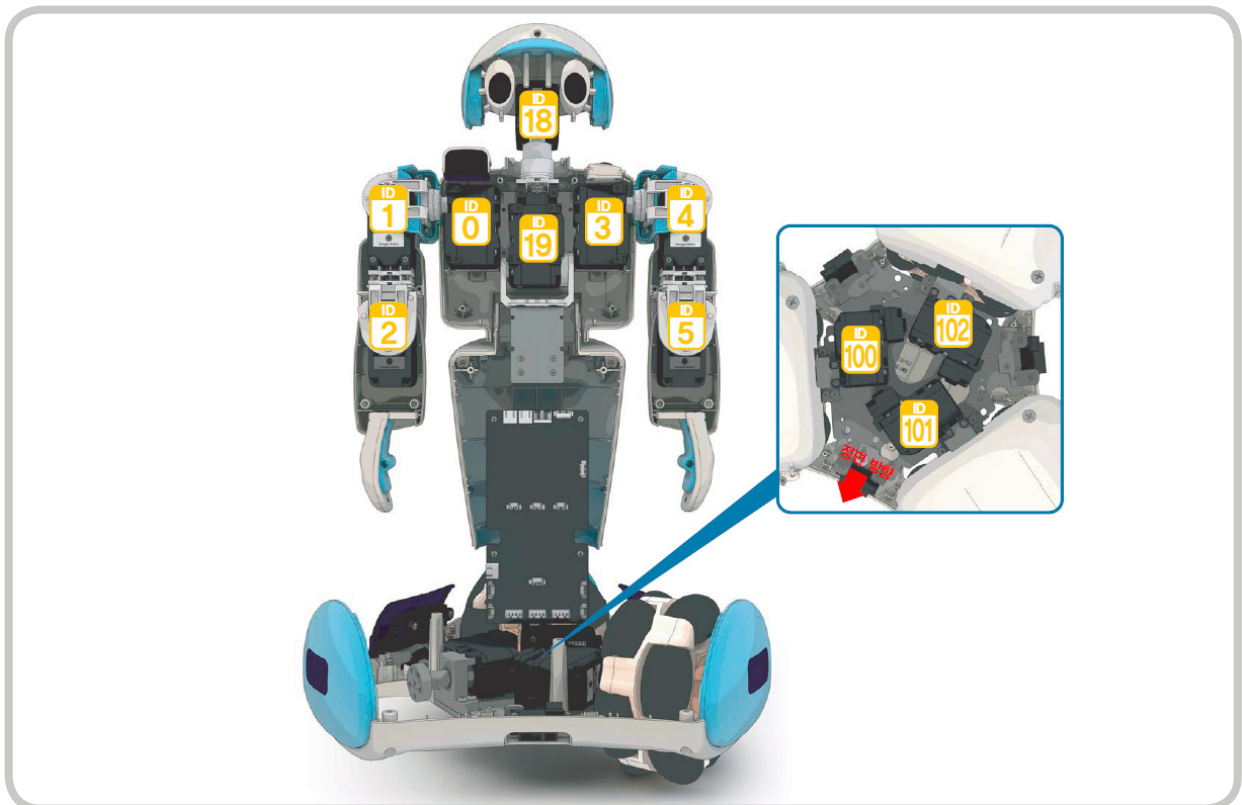
05 Compile, Download, Run

Click left compile button to compile the program. If there is no error, click right download button to download the program to MID. Click run button to run the program once download is complete.

4.2 Individual Control of Upper Body Motors

Example Description

It is usually more efficient to create and run a motion file in the robot to produce robot motion. However, if only one or two motors need to be controlled or to produce very simple robot motion, it maybe more efficient to control the motors directly rather than to create motion file. This example will produce robot motion by controlling the motors directly. Hovis Genie has 8 DRS-0101 servo motors in the upper body and 3 DRS-0102 motors in the omniwheel drive. Location and ID of each motor is shown in the diagram below.

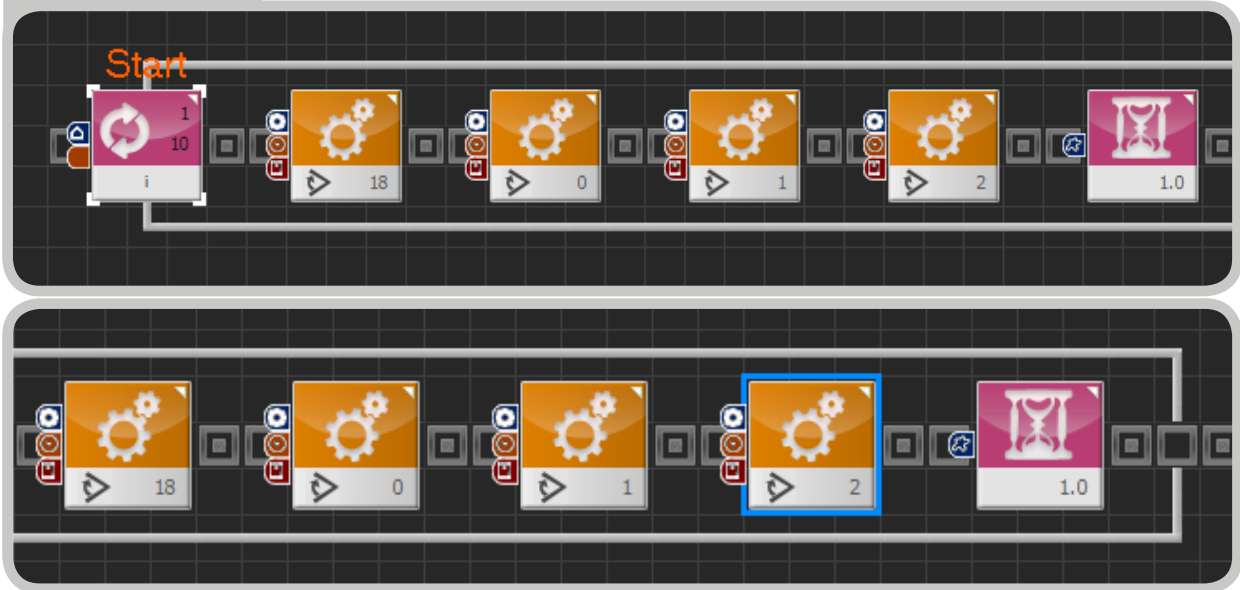


Each servo motor can be controlled by controlling the position and speed. Since all servo motors other than the motors with ID100~102 in the drivetrain are used as robot joints, it is irrelevant to control the speed. This example will control the motor position to repeatedly (10 times) extend the robot right arm from the attention position.

Entire Program

1. Loop – For loop (repeat 1~10 10 times)
2. Motor – Right arm and head servo motor position control(ID: 0, 1, 2, 18) – extend
3. Delay – Delay 1s
4. Motor – Right arm and head servo motor position control (ID: 0, 1, 2, 18) – Attention posture

Graphic

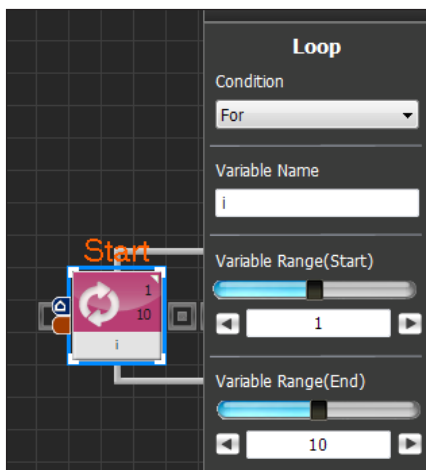


C-Like

```

1 short i
2 void main()
3 {
4     for( i = 1 ~ 10 )
5     {
6         jog( 412, 0, 18, 60 )
7         jog( 512, 0, 0, 60 )
8         jog( 412, 0, 1, 60 )
9         jog( 512, 0, 2, 60 )
10        delay( 1000 )
11        jog( 512, 0, 18, 60 )
12        jog( 235, 0, 0, 60 )
13        jog( 235, 0, 1, 60 )
14        jog( 512, 0, 2, 60 )
15        delay( 1000 )
16    }
17 }
    
```

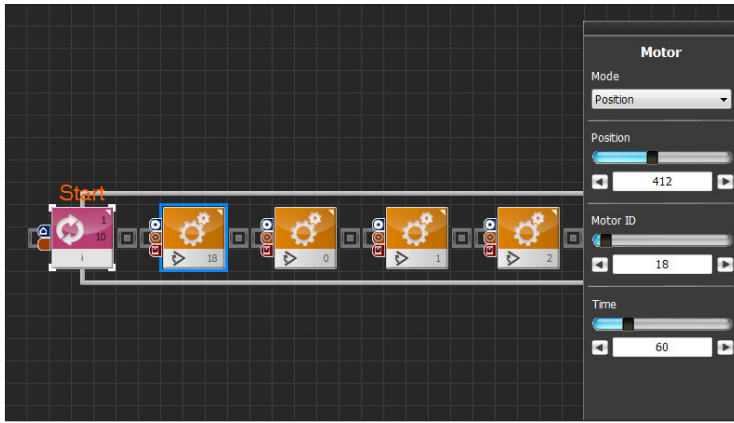
Step by Step



01 Loop

Select Loop module and dock to Start Point, Robot will repeat extending and lowering the right arm 10 times within the Loop. Set the loop property window values as follows.

- Condition : For
- Variable Name : i
- Variable Range(Start) : 1
- Variable Range(End) : 10



02 Motor

Use motor position control to extend the right arm. Right arm motor IDs are 0, 1, and 2. Head servo motor ID 18 is also controlled.

Add the Motor modules as shown in the diagram and enter the following values in the property window of each motor module .

Mode : Select position and speed control

Position : position value (0~1023)

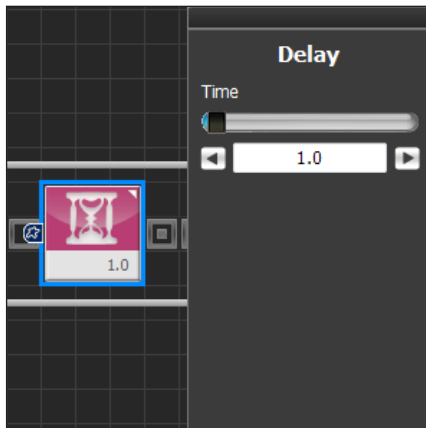
0 point (512)

Motor ID : motor ID

Time : 1 = 11,2ms

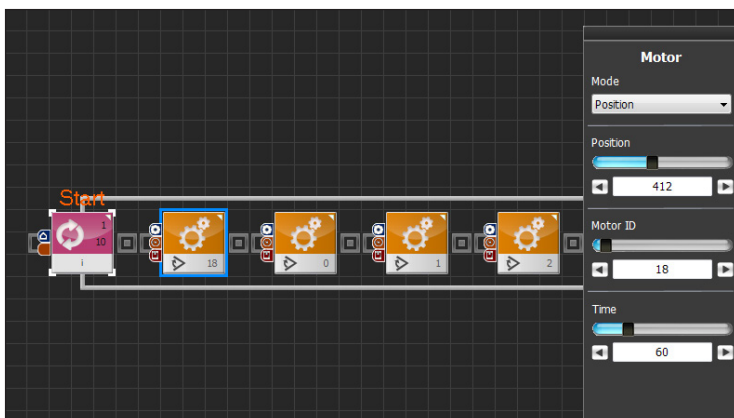
ex) 60 11,2ms = 672ms

Mode	Pos	Pos	Pos	Pos
Position	412	512	412	512
ID	18	0	1	2
Time	60	60	60	60



03 Delay

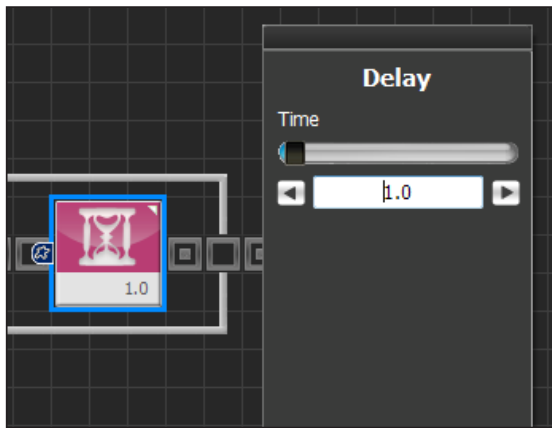
Pause program for 1s.



04 Motor

Use motor position control to lower the right arm. Right arm motor IDs are 0, 1, and 2. Head servo motor ID 18 is also controlled.

Mode	Pos	Pos	Pos	Pos
Position	512	235	235	512
ID	18	0	1	2
Time	60	60	60	60



05 Delay

Pause program for 1s.



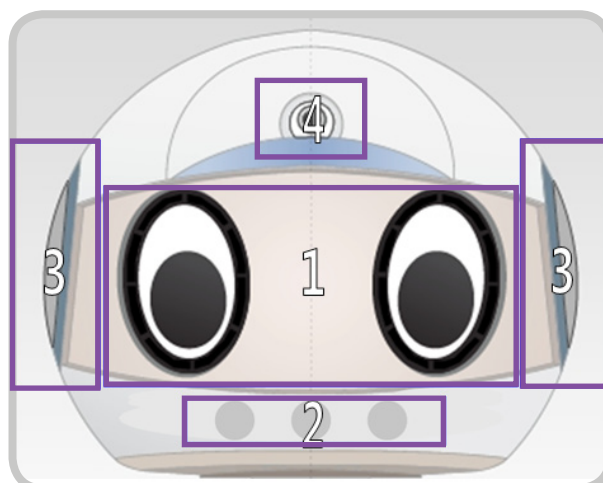
06 Compile, Download, Run

Click left compile button to compile the program. If there is no error, click right download button to download the program to MID. Click run button to run the program once download is complete.

4.3 Head LED Control

Example Description

Hovis Genie contains LED control board inside its head to control eye, ear, mouth, and forehead LEDs. Location of the LEDs are shown in the diagram below.



Location 1 contain eye LEDs. Each eye has 8 red and 8 blue LEDs for total of 32 LEDs in both eyes. Red and blue eye LEDs can be controlled individually. Turning on both red and blue LEDs simultaneously will produce purple color. Location 2 contain mouth LEDs comprised of 3 LEDs total. Location 3 contain ear LEDs with each ear comprised of 4 LEDs. Ear LEDs cannot be controlled individually. Location 4 contain brightness adjustable forehead LED.

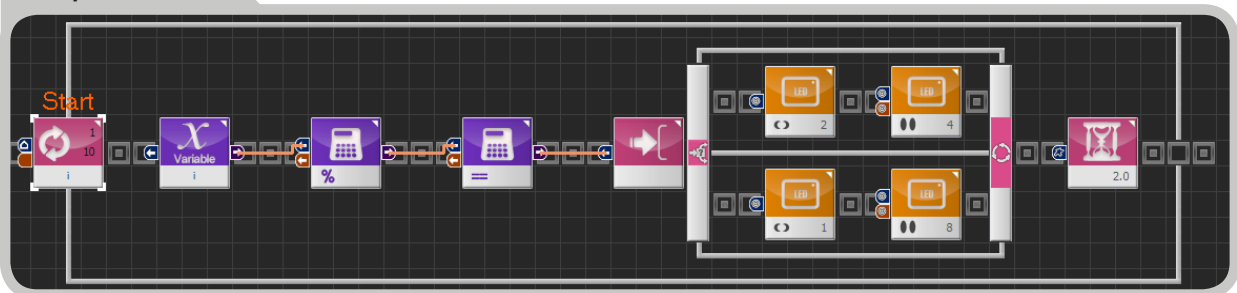
LED module provides functions for controlling eye, ear, mouth and forehead LEDs which can be used to express simulated robot emotion and also create visual effects.

This example will control Hovis Genie head LEDs. Program will repeat the loop 10 times and turn on the eye and ear LEDs alternately.

Entire Program

1. Loop – For loop i (repeat from 1~10 10 times)
2. If-Else – True if remainder after dividing i by 2 is 0, False otherwise
3. LED – Control eye and ear LEDs alternately
4. Delay – delay 2s

Graphic



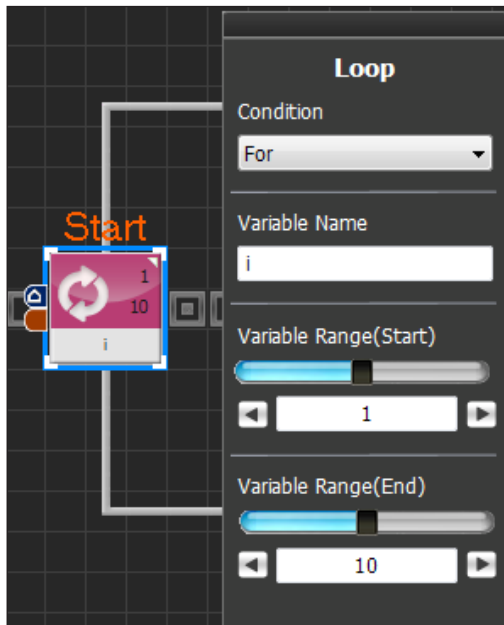
C-Like

```

1 short i
2 void main()
3 {
4     for( i = 1 ~ 10 )
5     {
6         if( ( i % 2 ) == 0 )
7         {
8             mid_led_ear( 2 )
9             mid_led_eye( 4, 2 )
10        }
11        else
12        {
13            mid_led_ear( 1 )
14            mid_led_eye( 8, 2 )
15        }
16        delay( 2000 )
17    }
18 }

```

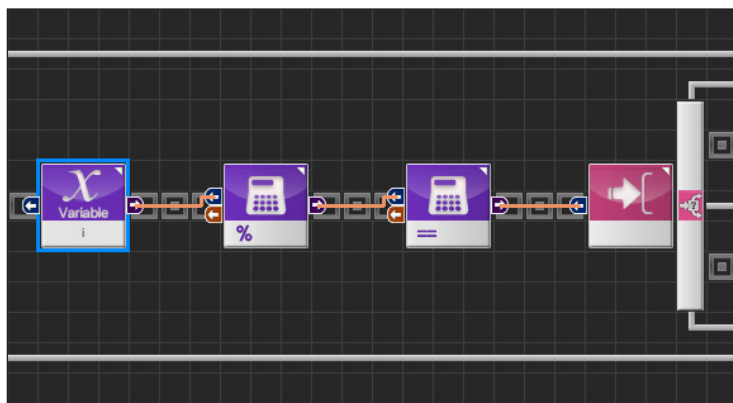
Step by Step



01 Loop

Select Loop module and dock to Start Point . Loop will repeat 10 times. Set loop module property window as follows.

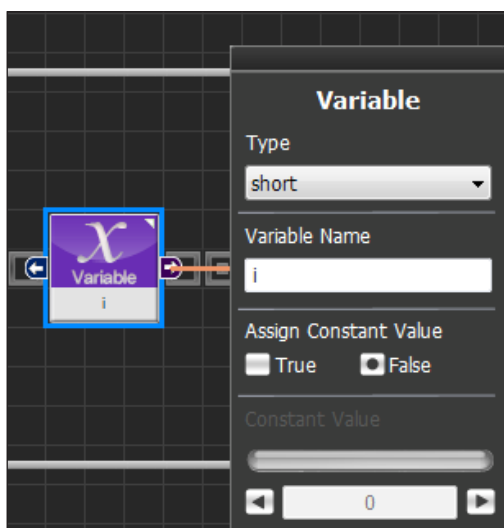
Condition : For
 Variable Name : i
 Variable Range(Start) : 1
 Variable Range(End) : 10



02 If-Else

If-Else module receives True or False value through the input pin.

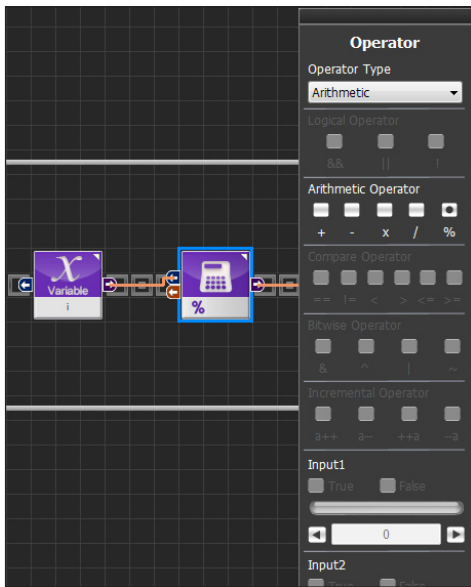
Compare the result of variable value i (which increases as loop in step 01 repeats) divided by 2 to 0 and enter the result in the input pin of If-Else module.



02-1 If-Else Conditional statement 1

Variable module
 Type: short
 Variable Name: i
 Assign Constant Value: False

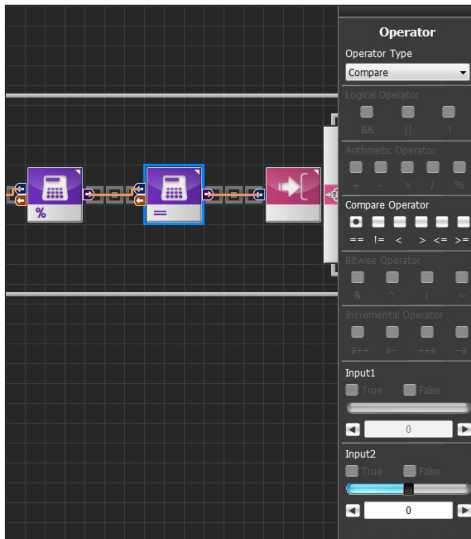
Set Type to short since loop variable i declared in step 01 is short type. Variable name is i since Loop variable i will be used. Set Assign Constant Value to False since assigned value will be substituted if value is set to True.



02-2 If-Else conditional statement 2

Operator module
 Operator Type: Arithmetic
 Arithmetic Operator: %
 Input2: 2

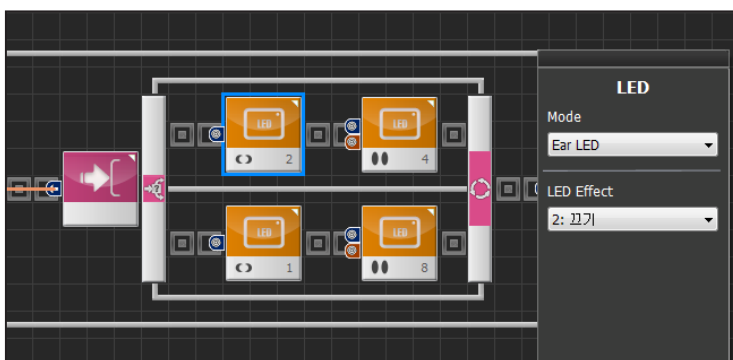
Enter the i value in step 02-1 as Input1, set Input2 to 2. Set Arithmetic Operator to % and enter the remainder of Input 1 value / Input 2 as the input pin value of the next module using the output pin.



02-3 If-Else conditional statement 3

Operator module
 Operator Type: Compare
 Arithmetic Operator: ==
 Input2: 0

Enter the result of operation in section 02-2 into Input1, Set Input2 to 0. Compare two values and output True if values are equal and False otherwise. Output value will be entered into the input pin of If-Else.



03 LED

Use LED module to turn on the eye and ear LEDs alternately.

Mode: Eye, Ear, Mouth, or Eye Brow select

LED Effect: select effect according to each mode

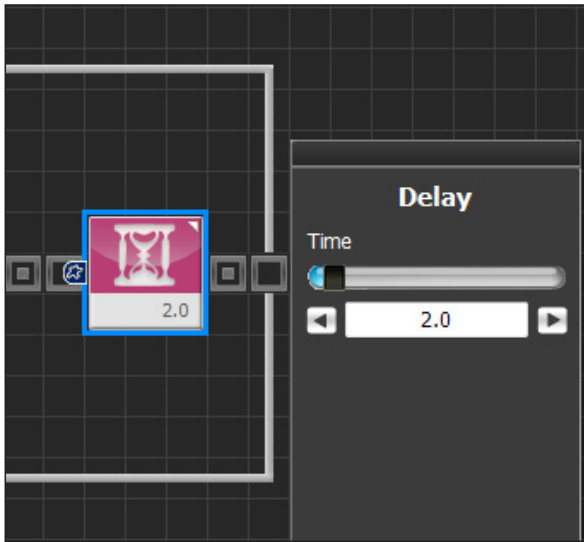
Set property window of each LED module as follows.

[TRUE]

	First	Second
Mode	Ear LED	Eye LED
LED Effect	2:Off	4:All Off

[FALSE]

	First	Second
Mode	Ear LED	Eye LED
LED Effect	1:On	8:On



04 Delay

Pause program for 2s. LED will turn on alternately every 2s.



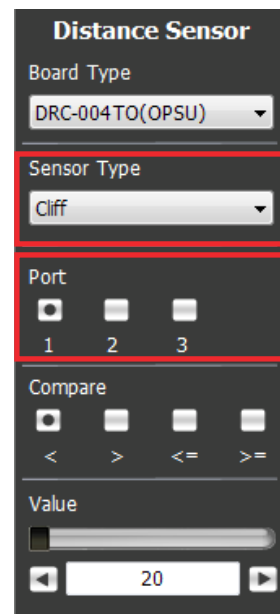
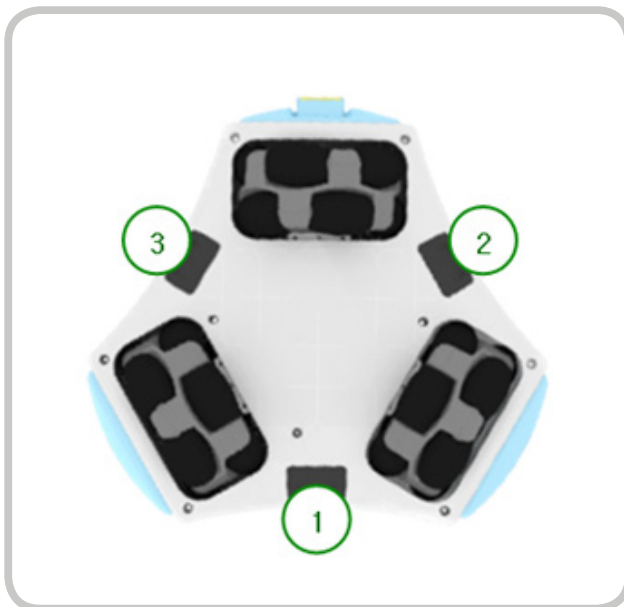
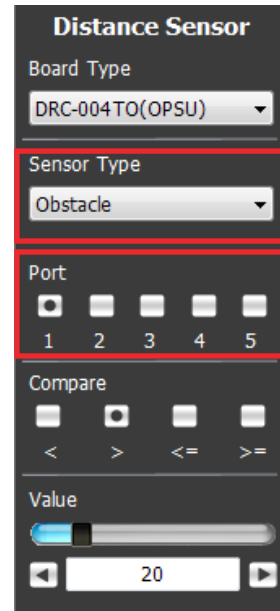
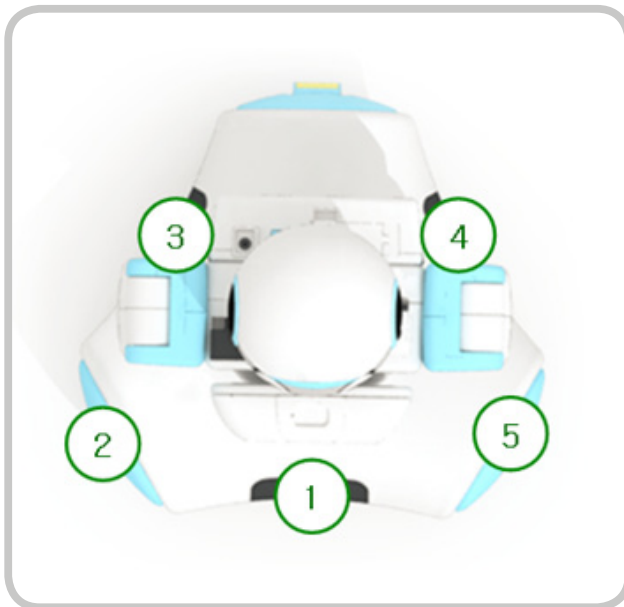
05 Compile, Download, Run

Click left compile button to compile the program. If there is no error, click right download button to download the program to MID. Click run button to run the program once download is complete.

4.4 Forward Obstacle Detection

Example Description

Hovis Genie uses 8 PSD sensors to recognize the surrounding environment. 5 sensors are used as distance sensors and remaining 3 are used as bottom sensors. Distance Sensor module property window settings for each sensor are as follows.



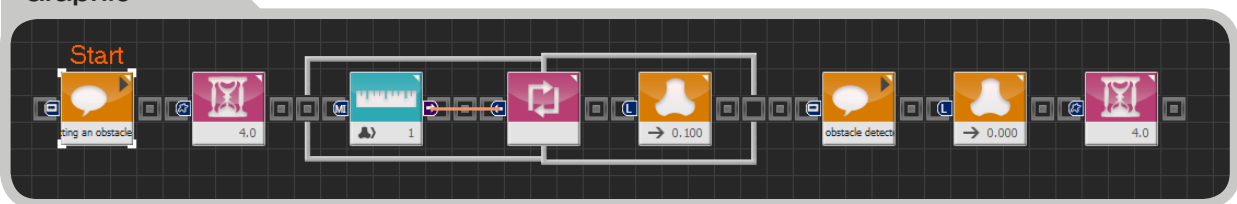
PSD sensors are capable of detecting obstacles within 80cm distance.

This example program will move the robot forward, stop and use voice to announce obstacle detection when it detects an obstacle.

Entire Program

1. TTS – Starting obstacle detection voice announcement
2. Delay – 4s delay
3. While – Infinite loop if input pin is TRUE
 - Condition : Distance Sensor – True if Obstacle(1) is larger than 20cm, False if smaller.
 - When TRUE : Wheel – Move robot forward
 - When FALSE : Exit loop
4. TTS – Obstacle detected voice announcement
5. Wheel – Stop wheels
6. Delay – Delay 4s

Graphic

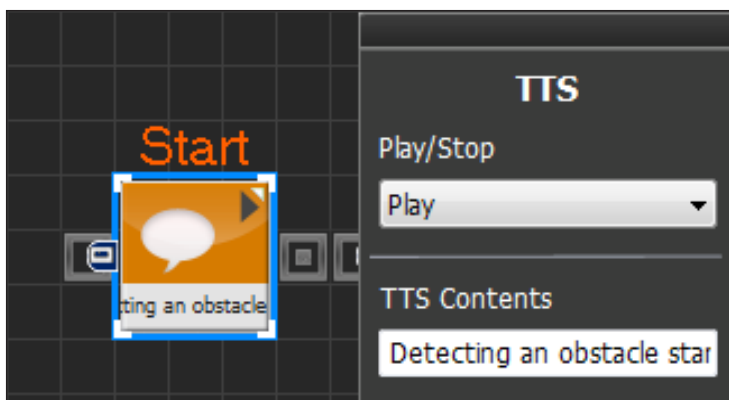


C-Like

```

1 void main()
2 {
3     mid_tts_play( "Detecting an obstacle start." )
4     delay( 4000 )
5     while( ( OPSU_ObstaclePSD1 > 20 ) )
6     {
7         mid_wheel_linear( 0.100, 0, false )
8     }
9     mid_tts_play( "An obstacle detected." )
10    mid_wheel_linear( 0.000, 0, false )
11    delay( 4000 )
12 }
    
```

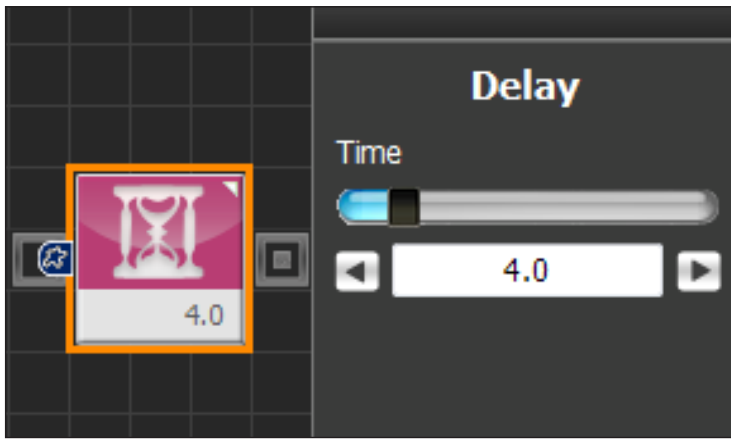
Step by Step



01 TTS

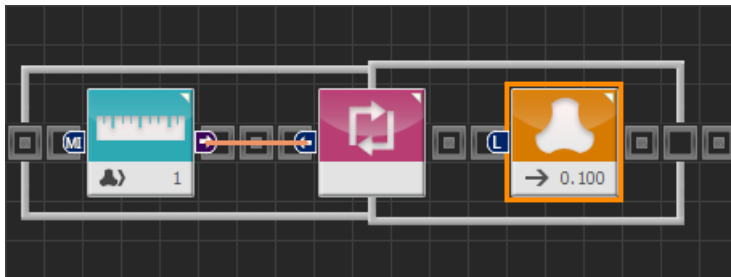
Output TTS message announcing detection of forward obstacles.

- Play/Stop : Play
- TTS Contents
"Detecting forward obstacles."



02 Delay

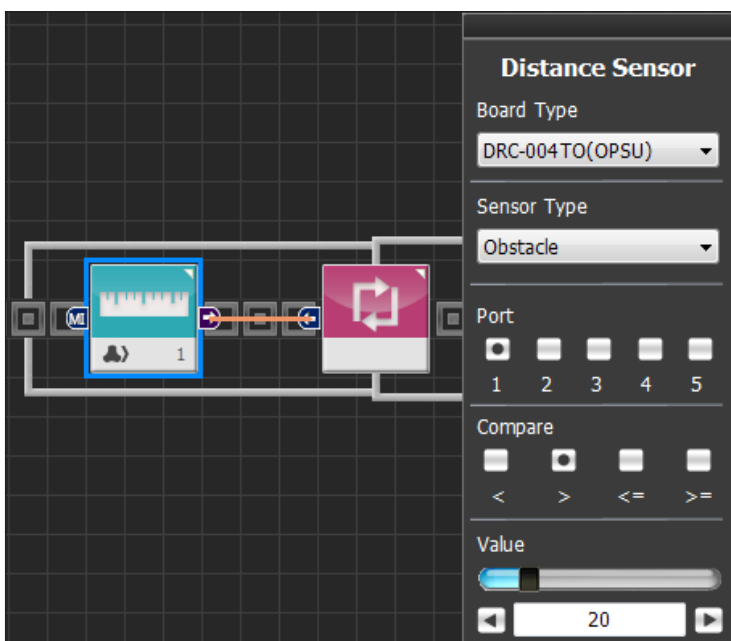
TTS module does not wait until the voice announcement ends. In otherwords, next module is being processed while the TTS announcement is still in progress. Delay the play to provide enough time to listen to the TTS announcement.



03 While

While module repeats the loop when input pin value is TRUE and exits the loop when value is FALSE.

Distance Sensor module provides the value to the input pin. Robot moves forward when the value is TRUE.

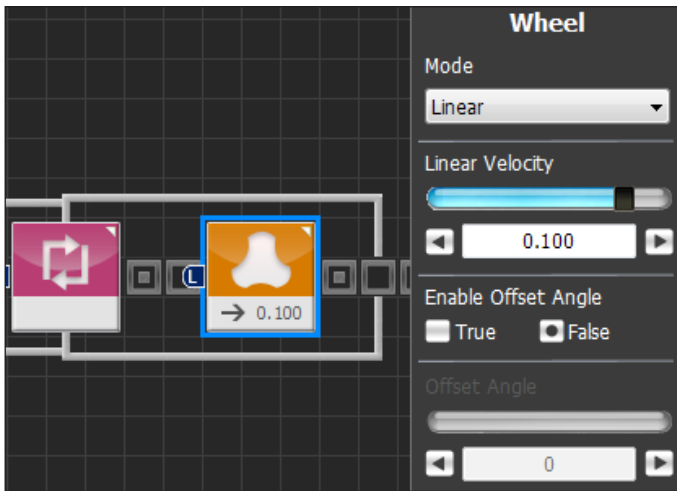


03-1 While Module Condition

Distance Sensor module reads the selected PSD sensor value.

- Board Type: DRC-004TO(OPSU)
- Sensor Type: Obstacle
- Port : 1
- Compare : >
- Value : 20

This module reads the forward PSD sensor value and enters TRUE if the distance value is larger than 20cm and FALSE if less than 20cm to the output pin.

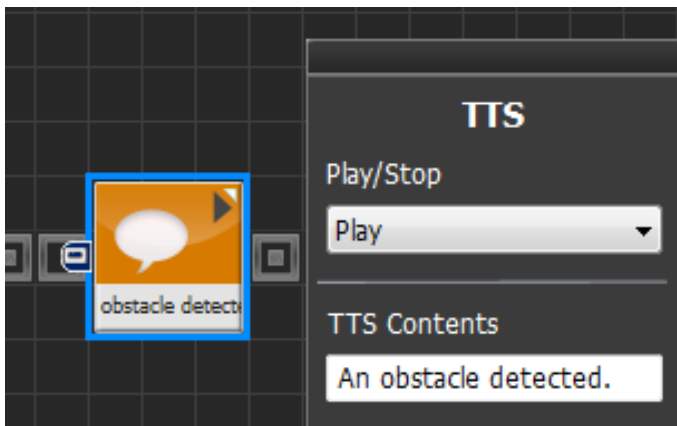


03-2 While Module Loop

Wheel module within the While module loop is repeated until the condition in section 03-1 is NOT FALSE.

Wheel module moves the robot forward. Wheel module property window values are set as follows.

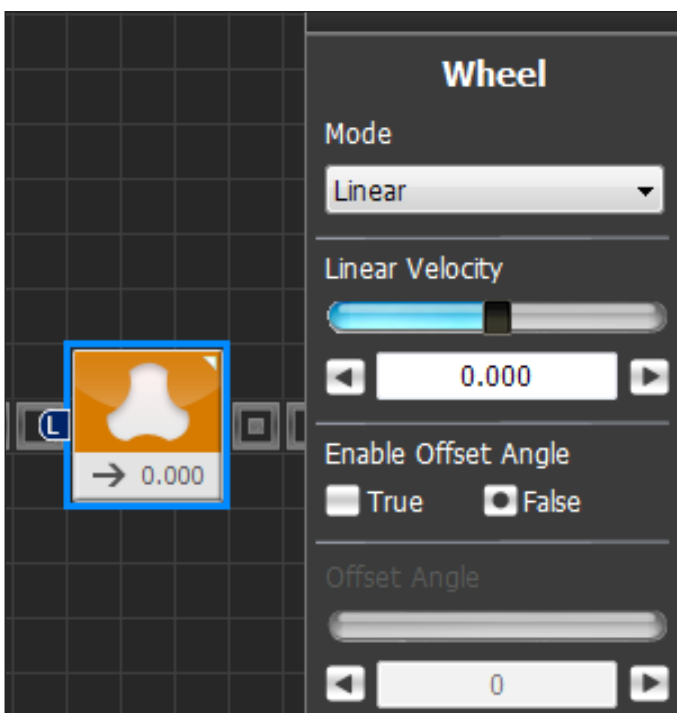
- Mode : Linear
- Linear Velocity: 0.100
- Enable Offset Angle : False



04 TTS

Output TTS announcement announcing obstacle detection.

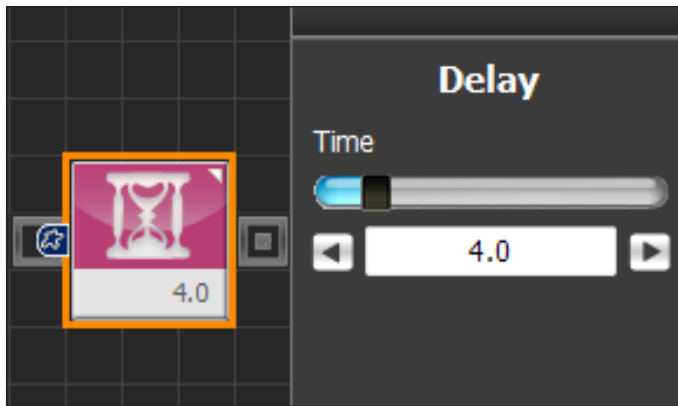
- Play/Stop : Play
- TTS Contents
"Obstacle detected"



05 Wheel

This Wheel module stops the robot drivetrain. Wheel module property window values are set as follows.

- Mode : Linear
- Linear Velocity: 0.000
- Enable Offset Angle : False



06 Delay

TTS module does not wait until the voice announcement ends. In other words, next module is being processed while the TTS announcement is still in progress. Delay the play to provide enough time to listen to the TTS announcement.



07 Compile, Download, Run

Click left compile button to compile the program. If there is no error, click right download button to download the program to MID. Click run button to run the program once download is complete.

4.5 Touch Sensor and Sound Output

■ Example Description

Touch sensors are used to interact with the user. Hovis Genie has total of 3 touch sensors located at each palm and on the head.

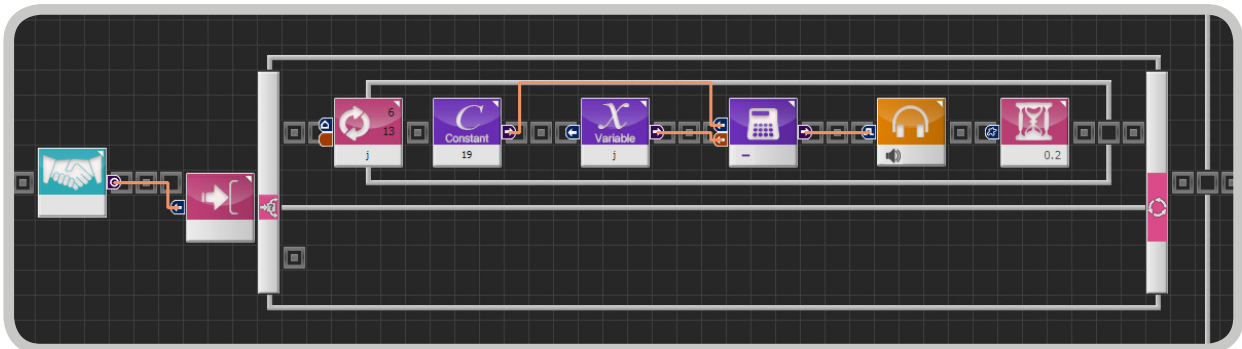
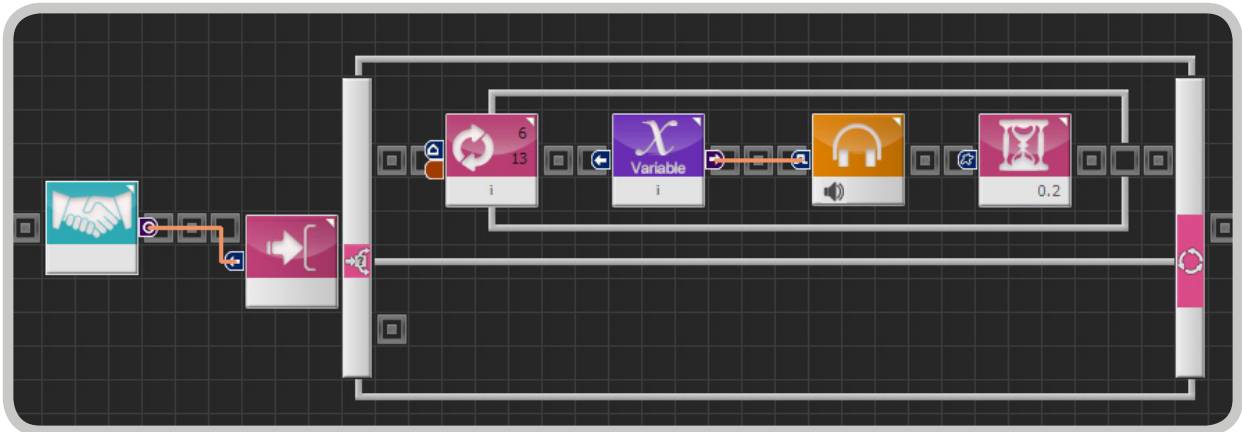
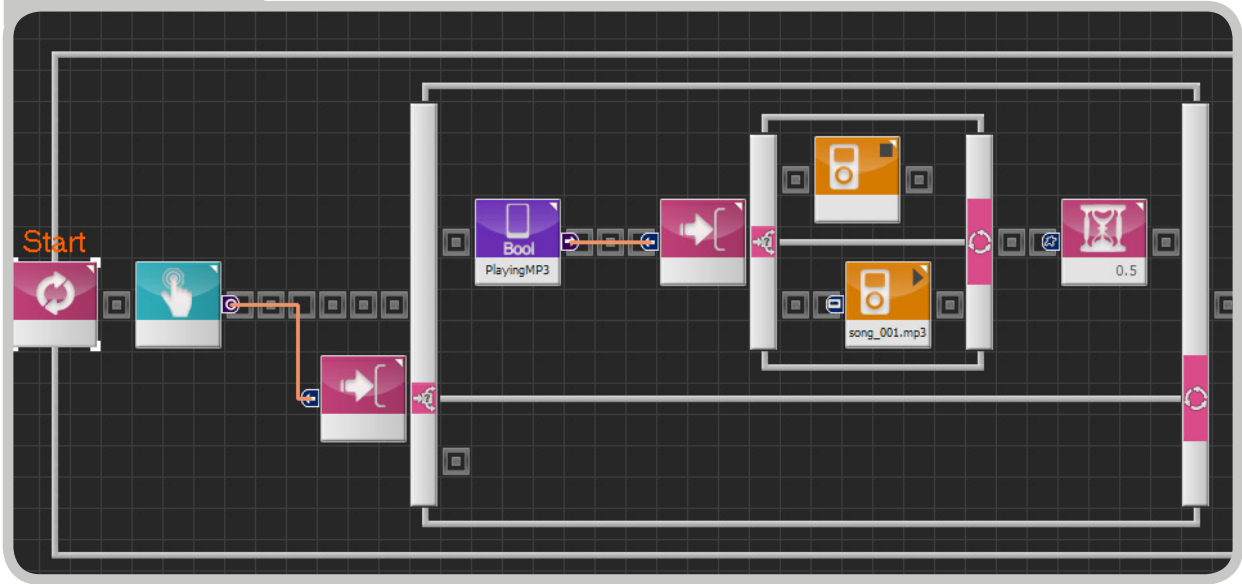


This example program will use 3 touch sensors to output sound. Hovis Genie will start to play MP3 file when it detects touch to the head or stop the play if the file is already being played. Also, Hovis Genie will play “DoReMiFaSolLaSiDo” and “DoSiLaSolFaMiReDo” when the palms are touched.

■ Entire Program

1. Loop – Infinite Loop
2. Touch Sensor – Head touch detection
 - When TRUE : Use MID RAM(PlayingMP3) variable to check MP3 play status
 - When PlayingMP3 is TRUE : Stop MP3 play
 - When PlayingMP3 is FALSE : Start MP3 play
3. Hand Touch Sensor(Right) – Detect right hand touch
 - When TRUE : Play “DoReMiFaSolLaSiDo” using sound module
4. Hand Touch Sensor(Left) – Detect left hand touch
 - When TRUE : Play “DoSiLaSolFaMiReDo” using sound module

Graphic

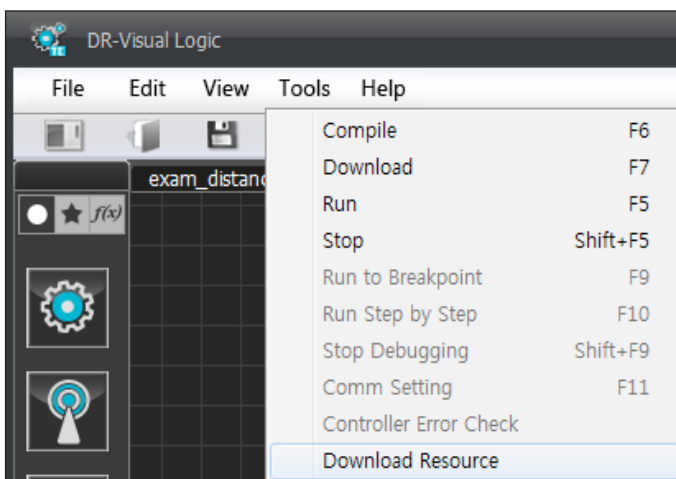


```

1  snort(i,j)
2  void main()
3  {
4      while( true )
5      {
6          if( ( MPSU_TouchStatus ) )
7          {
8              if( MID_PlayingMP3 )
9              {
10                 mid_mp3_stop()
11             }
12             else
13             {
14                 mid_mp3_play( "song_001.mp3" )
15             }
16             delay( 500 )
17         }
18         else
19         {
20         }
21         if( ( SERVO_GPIO1[2]==0 ) )
22         {
23             for( i = 6 ~ 13 )
24             {
25                 mid_sound( i )
26                 delay( 200 )
27             }
28         }
29         else
30         {
31         }
32         if( ( SERVO_GPIO1[5]==0 ) )
33         {
34             for( j = 6 ~ 13 )
35             {
36                 mid_sound( ( 19 - j ) )
37                 delay( 200 )
38             }
39         }
40         else
41         {
42         }
43     }
44 }

```

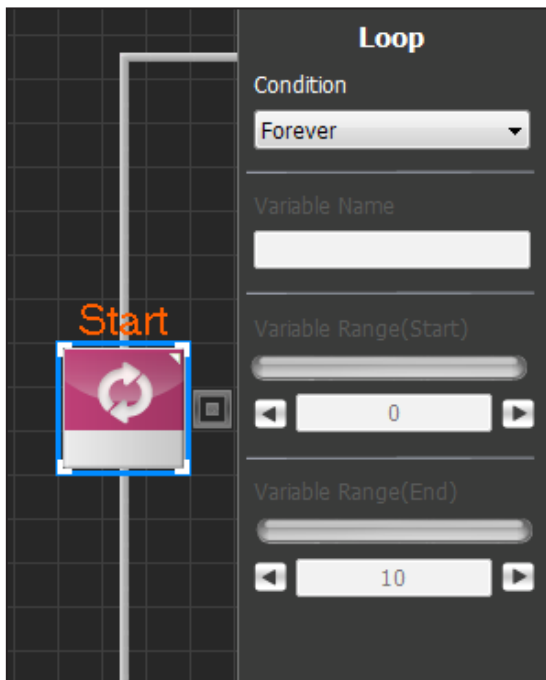
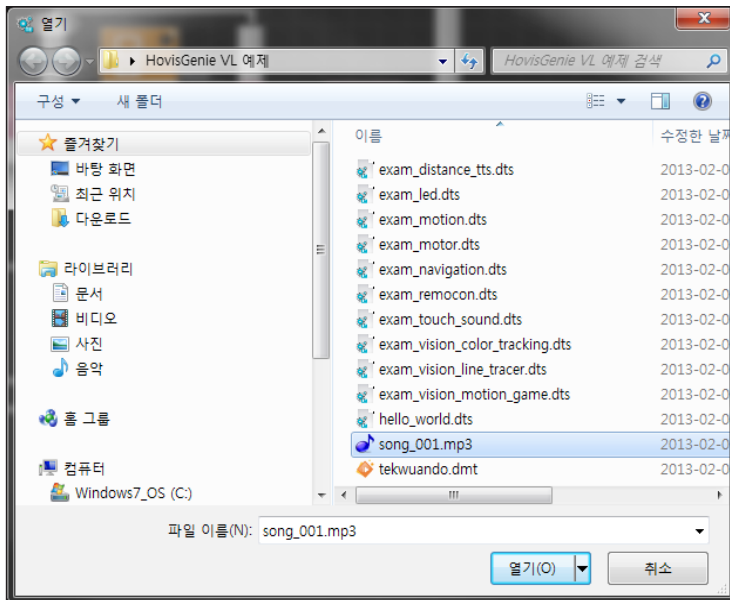
Step by Step



01 Preparation

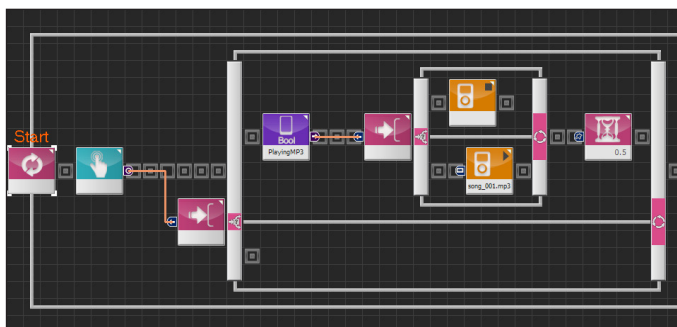
Connect DR-Visual Logic and MID.
Download MP3 file to MID.

- Menu > Tool > Click resource download
- Select and download MP3 file to be played.



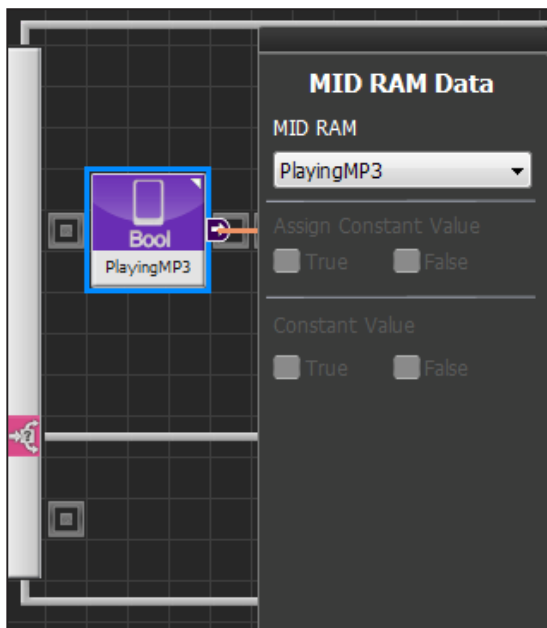
02 Loop

Select loop module and dock to Start Point. Set the Condition in loop module property window to Forever.



03 Touch Sensor (Head touch detection)

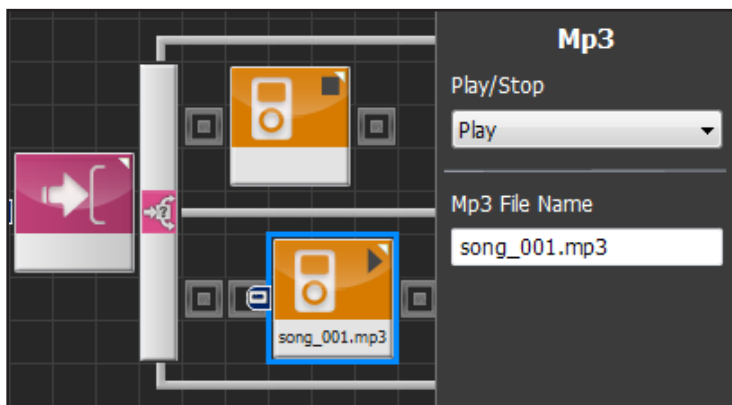
Place Touch Sensor module and connect the Touch sensor module output pin and If-Else module output pin using connection line.



PlayingMP3 variable in MID RAM Data module will output TRUE if MP3 is playing and output FALSE otherwise.

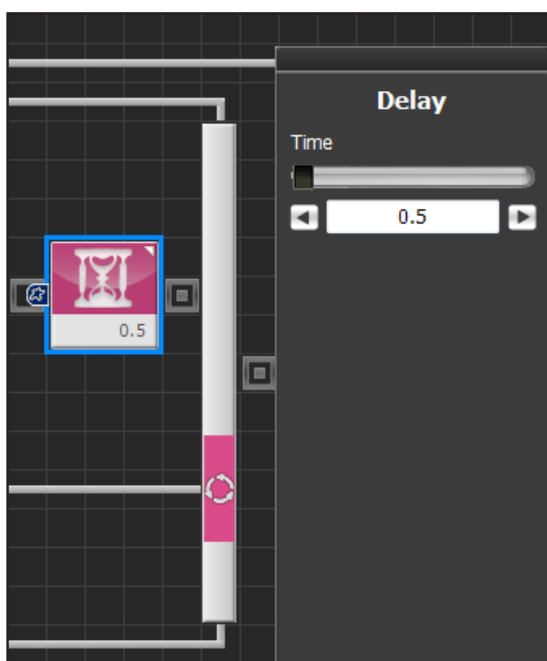
Place If-Else module and MP3 module.

Set Play/Stop property of the MP3 module in TRUE section to Stop. Stop will stop the current MP3 file being played regardless of the file.

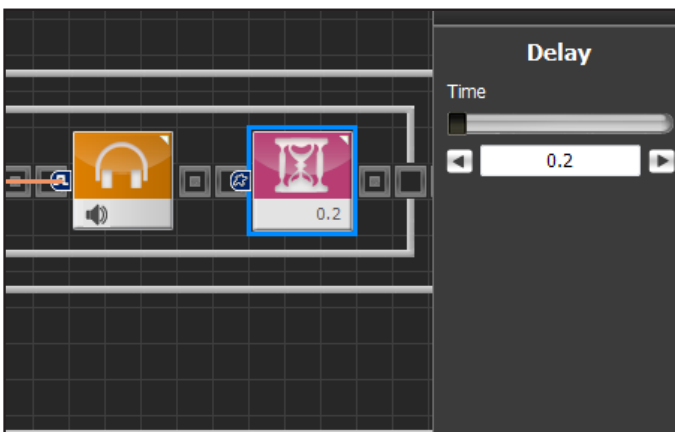
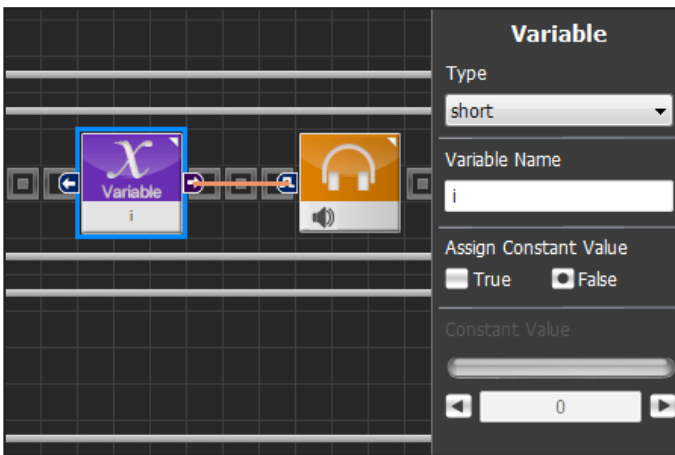
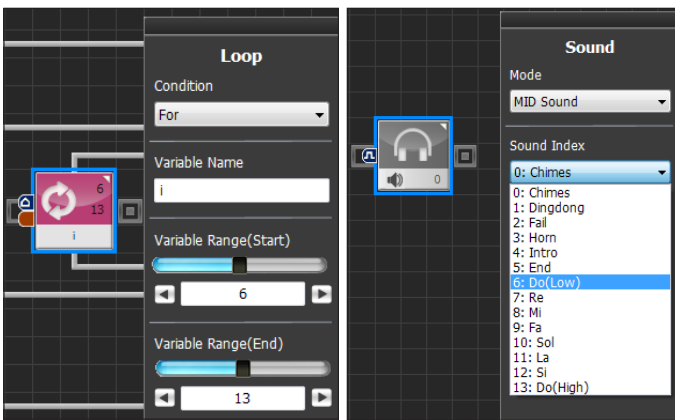
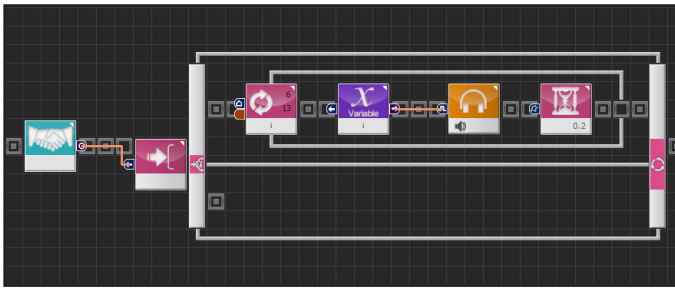


Enter the the name of the MP3 file to play into the property of the MP3 module in the FALSE section.

- Play/Stop : Play
- Mp3 File Name : song_001.mp3



Place the Delay module. Without Delay module, Hovis Genie will recognize single head touch as double touch and stop the MP3 play as soon as it starts. This is due to the fact Touch Sensor module is located within the Loop module.



04 Hand Touch Sensor (Right hand)

Place Hand Touch Sensor and enter Right in the property window to detect touch to the right hand.

Hand Touch Sensor will output TRUE when right tact switch is pressed and output FALSE otherwise.

Connect Hand Touch Sensor output pin and If-Else module output pin using connector line.

Loop module required to play “DoReMiFaSolLaSiDo” when right hand is touched.

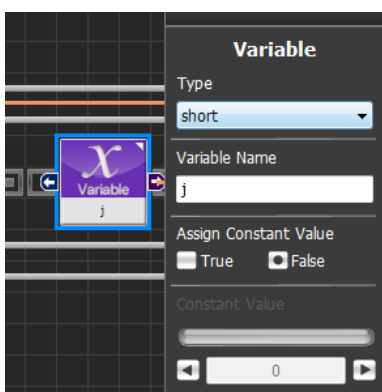
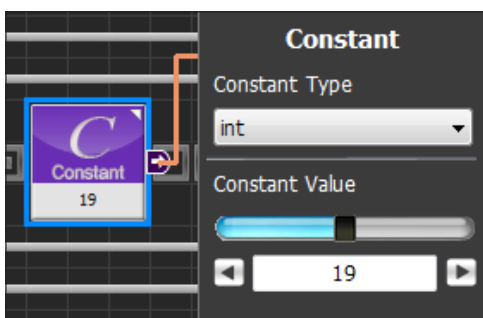
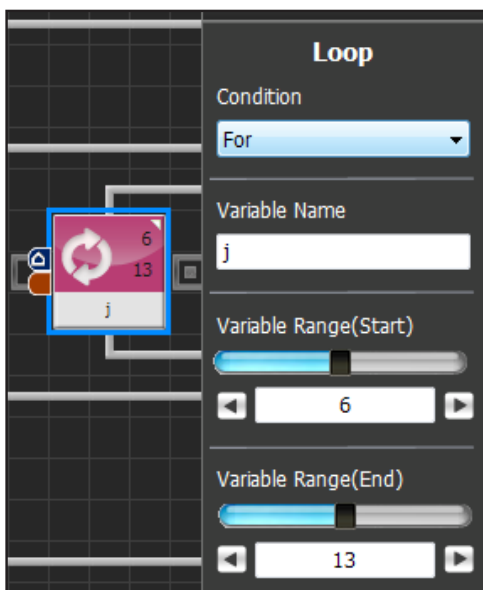
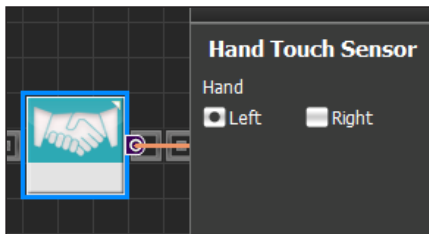
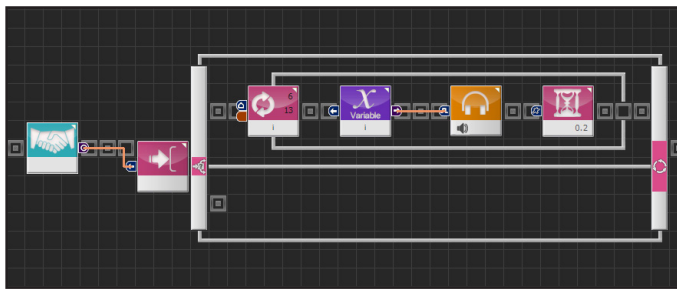
Add Loop module.

- Condition: For
- Variable Name: i
- Variable Range(Start): 6
- Variable Range(End): 13

Range is 6~13 because Sound module piano note Sound Index is 6~13.

Use Variable module to use the i value assigned in the loop module. Set Type property to Short since i variable used in the Loop module is short type.

Use Delay module to provide 0.2s interval between the notes. Without the interval, played notes would become almost indistinguishable.



05 Hand Touch Sensor (Left hand)

Place Hand Touch Sensor and enter Left in the property window to detect touch to the left hand.

Hand Touch Sensor will output TRUE when left tact switch is pressed and output FALSE otherwise.

Connect Hand Touch Sensor output pin and If-Else module output pin using connector line.

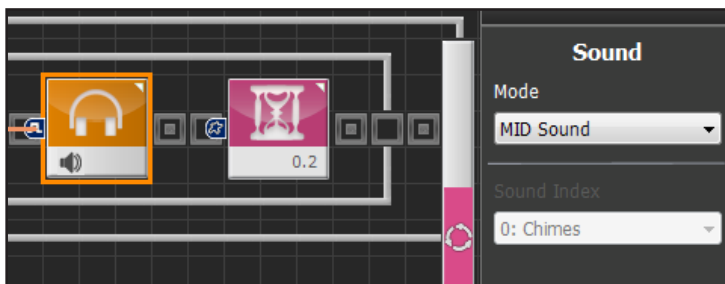
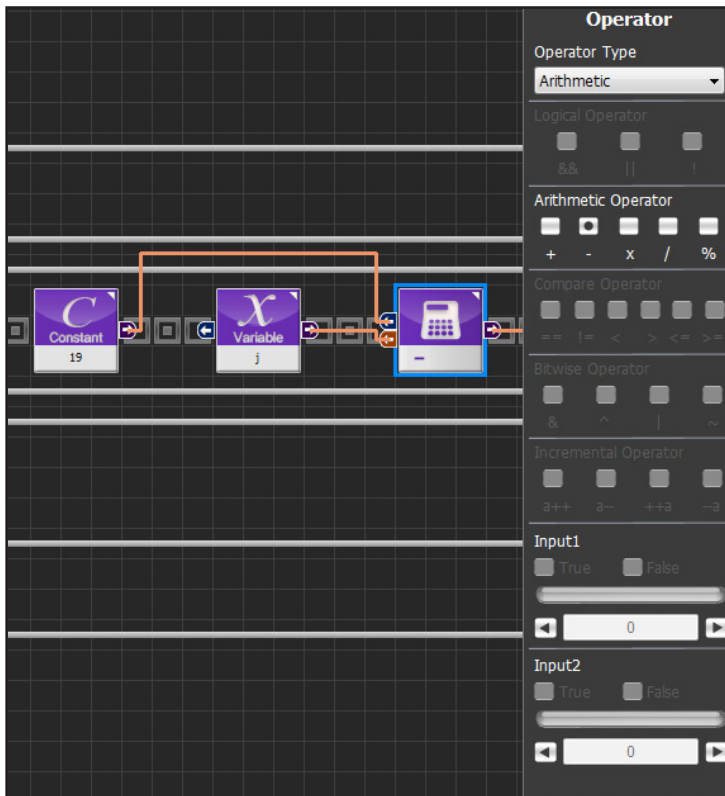
Loop module required to play "DoSiLaSolFaMiReDo" when left hand is touched.

Add Loop module.

- Condition: For
- Variable Name: j
- Variable Range(Start): 6
- Variable Range(End): 13

In case of right hand Sound module, values from 6 to 13 is entered in the sound module as loop is repeated. However, for the left hand, values from 13 to 6 needs to enter the sound module.

To convert 6~13 to 13~6, deduct value of j variable used in the loop module from 19. For example, if $j = 6$, $19 - 6 = 13$, if $j = 13$, $19 - 13 = 6$.



Enter the calculated value into the sound module using the connector line.

Use Delay module to provide 0.2s interval between the notes. Without the interval, played notes would become almost indistinguishable.

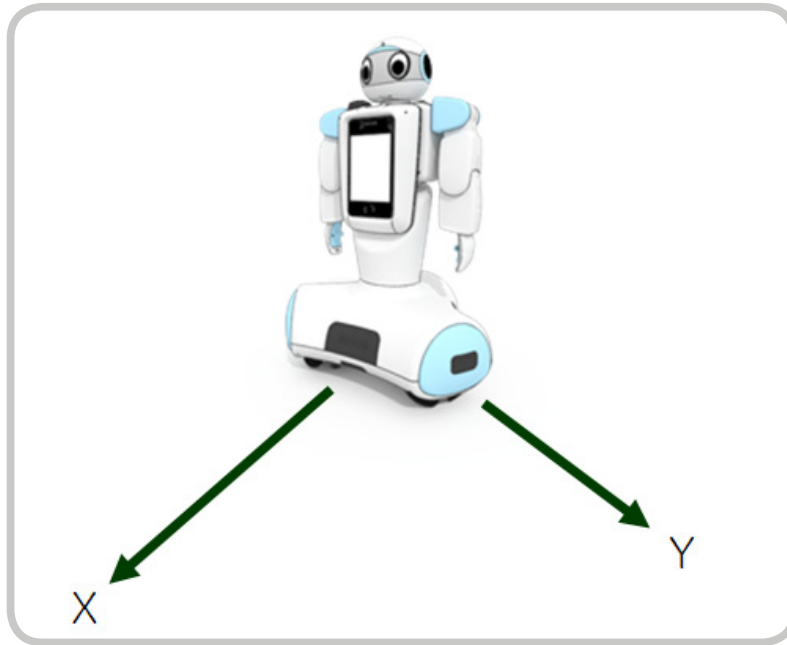
06 Compile, Download, Run

Click left compile button to compile the program. If there is no error, click right download button to download the program to MID. Click run button to run the program once download is complete.

4.6 Navigation Movement

Example Description

Hovis Genie navigation movement refers to movement following the waypoint coordinates in 2 dimensional plane with robot as the center of reference and X axis as the front. Unit of measure for waypoint coordinate is in m. Diagram below shows the robot coordinate axis.



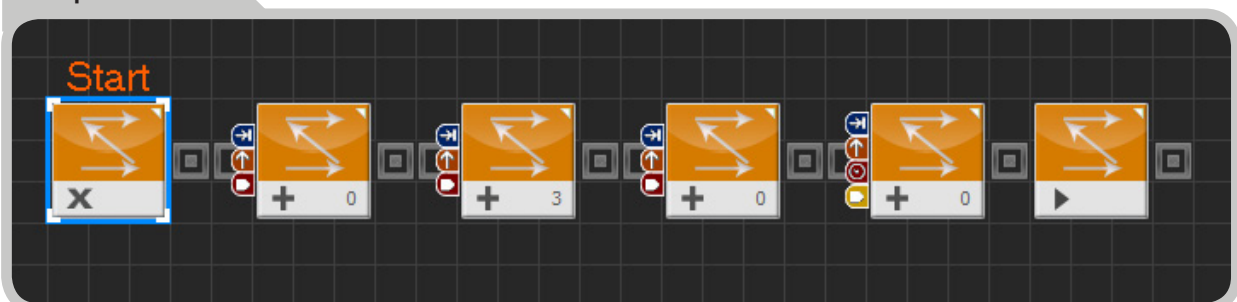
Navigation movement uses the wheel rotation count to calculate the coordinates but this method is prone to error which accumulates as movement distance increases. Due to this reason, certain amount of error is expected in the navigation movement provided by DR-Visual Logic navigation module. As a reference, robot technology allowing the robot to make automatic course correction to provide more precise navigation is one of the area that is being researched vigorously.

This example program will make the robot navigate a diamond pattern with current location as reference.

Entire Program

1. Navigation – Initialization
2. Navigation – Add waypoint
3. Navigation – Start Navigation movement

Graphic

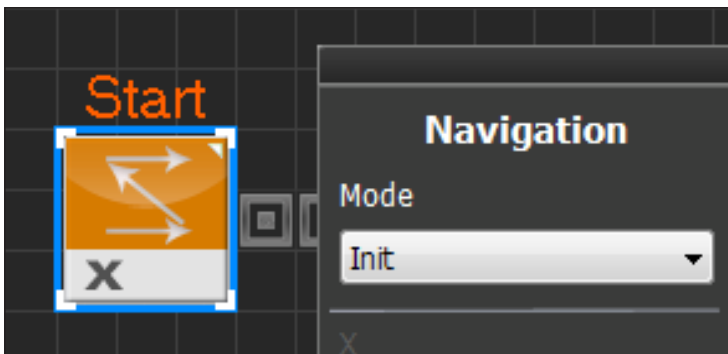


C-Like

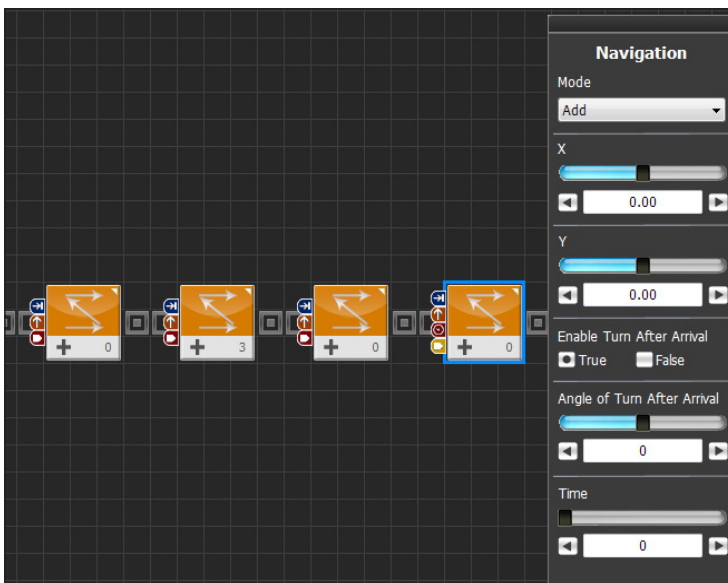
```

1 void main()
2 {
3     mid_navi_init()
4     mid_navi_add( 0.50, 0.50, 0, false, 0 )
5     mid_navi_add( 1.00, 0.00, 0, false, 3 )
6     mid_navi_add( 0.50, -0.50, 0, false, 0 )
7     mid_navi_add( 0.00, 0.00, 0, true, 0 )
8     mid_navi_start()
9 }

```

Step by Step**01 Navigation Initialization**

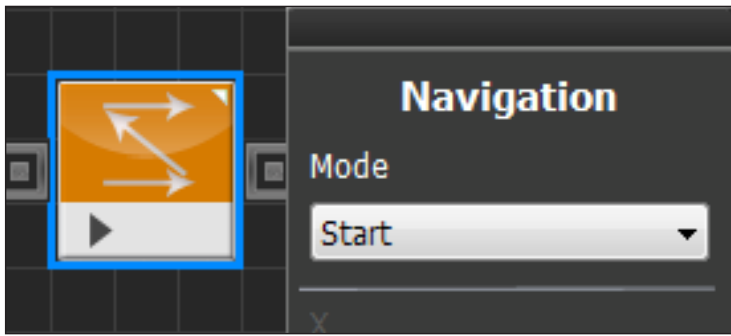
Add Navigation module and set the Mode in the property window to Init. When navigation is initialized, robot creates virtual coordinate with front of current location as X axis. In other words, current location is set a (0,0).

**02 Add Navigation Waypoints**

Add waypoints needed for the robot to make diamond patterned movement. Add 4 Navigation modules and set properties for each module.

	Mode	X	Y	E.	A.	Time
1	Add	0.5	0.5	F	-	0
2	Add	1.0	0.0	F	-	3
3	Add	0.5	-0.5	F	-	0
4	Add	0.0	0.0	T	0	0

- E. (Enable Turn After Arrival)
- A. (Angle of Turn After Arrival)
Navigation Property Window
- X: X coordinate (Unit:m)
- Y: Y coordinate (Unit:m)
- Enable Turn After Arrival
Decide whether to assign the direction robot is facing after arriving at the waypoint.
- Angle of Turn After Arrival Set the direction robot to face after arriving at the waypoint, (-180~180)
- Time Wait time after arriving at the waypoint.



03 Beign Navigation

Add Navigation module and set Mode to Start.



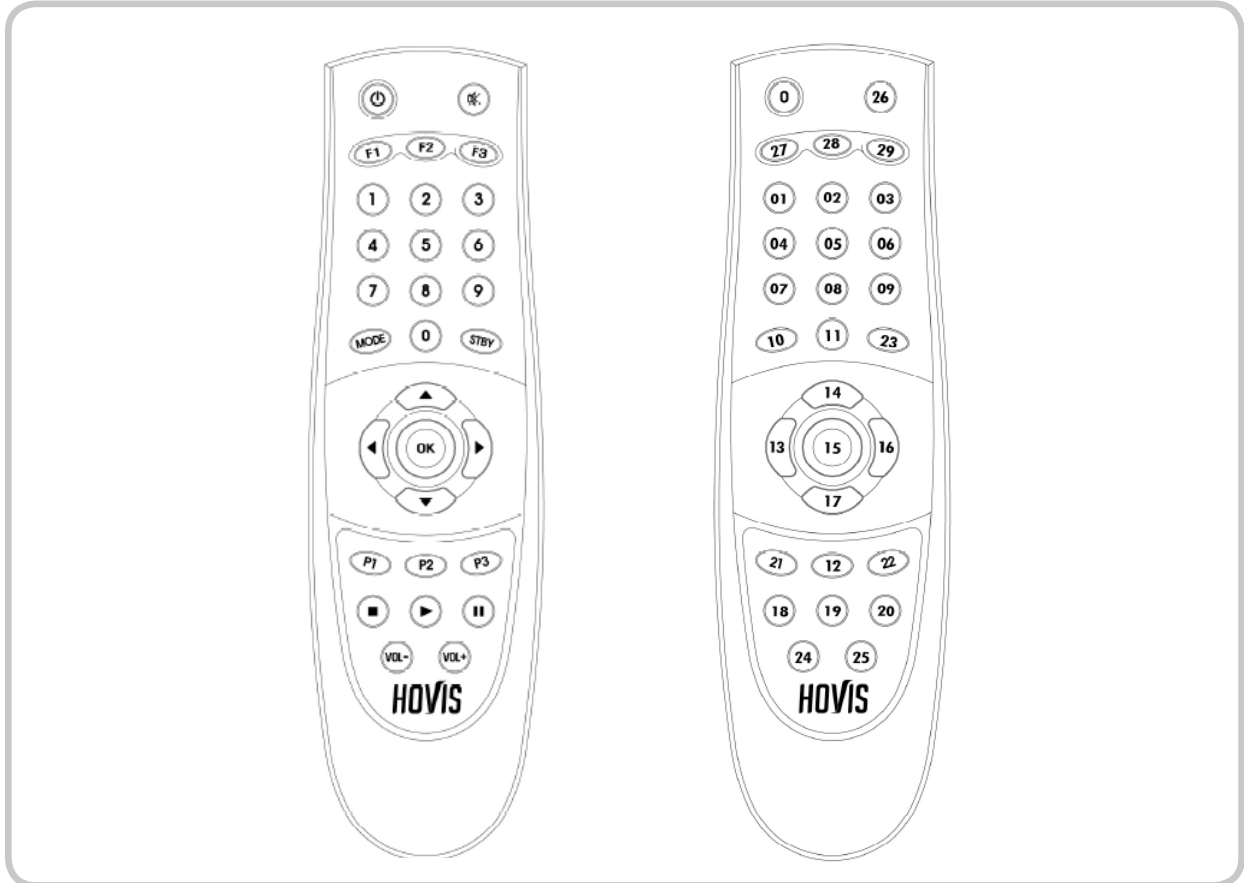
04 Compile, Download, Run

Click left compile button to compile the program. If there is no error, click right download button to download the program to MID. Click run button to run the program once download is complete.

4.7 Remote Control Drive

■ Example Description

Hovis Genie remote control and IRReceive module can be used to create various programs. Numbers assigned to remote control buttons are as follows.



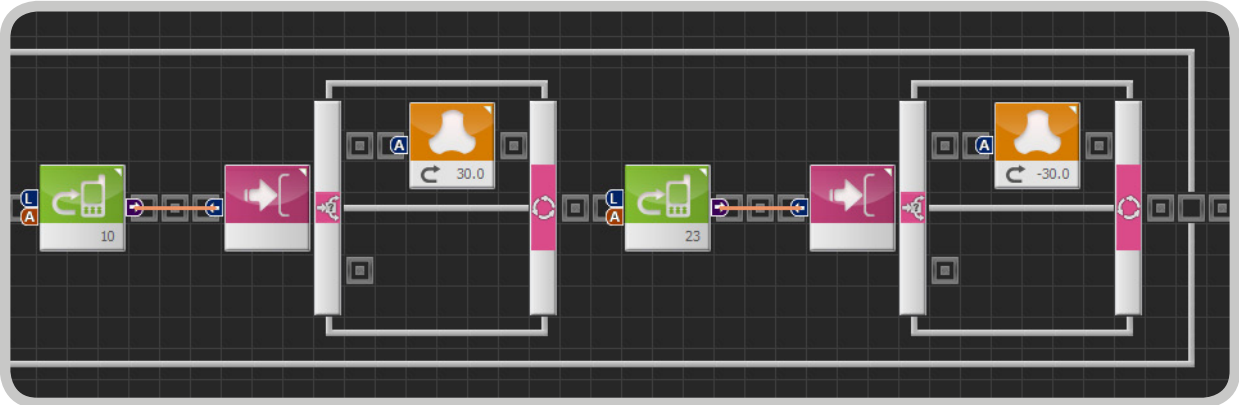
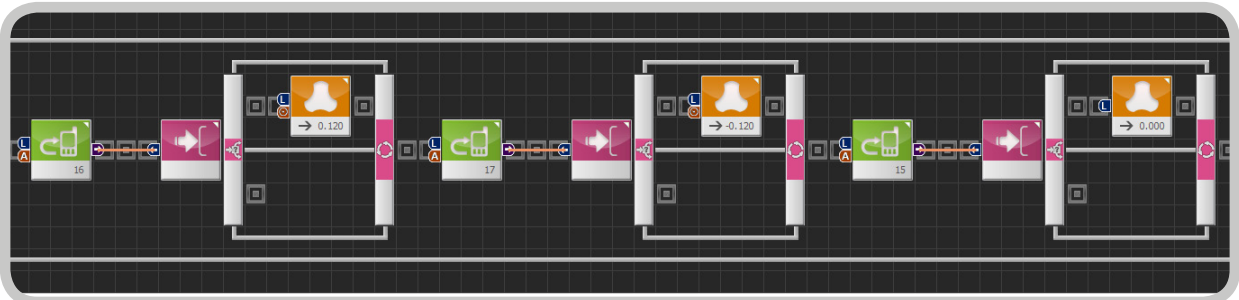
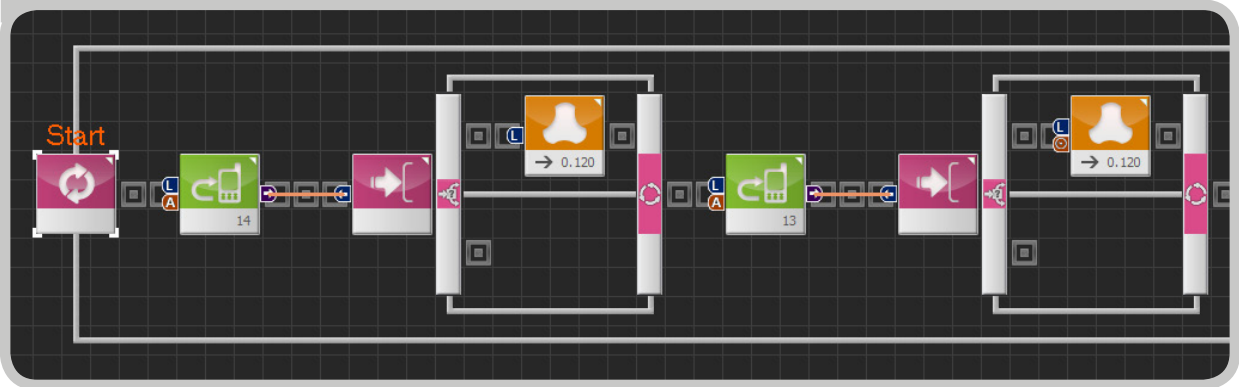
This example program uses the remote control to control the robot drivetrain.

■ Entire Program

1. Loop – Infinite Loop
2. IRReceive – Confirm data reception from applicable remote control.
3. Wheel – Control robot drivetrain in accordance with the data received from the applicable remote control.

Remote Control Data	Drivetrain
14	Forward
13	Left sideway movement
16	Right sideway movement
17	Backward
10	Left rotation
23	Right rotation
15	Stop

Graphic



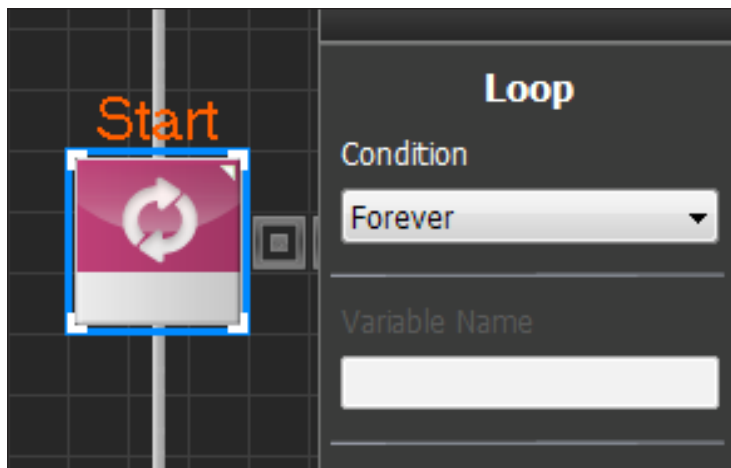
C-Like

```

1 void main()
2 {
3     while( true )
4     {
5         if( ( MPSU_RmcLength >= 8 && MPSU_RmcData == 14 ) )
6         {
7             mid_wheel_linear( 0.120, 0, false )
8         }
9         else
10        {
11        }
12        if( ( MPSU_RmcLength >= 8 && MPSU_RmcData == 13 ) )
13        {
14            mid_wheel_linear( 0.120, 90, true )
15        }
16        else
17        {
18        }
19        if( ( MPSU_RmcLength >= 8 && MPSU_RmcData == 16 ) )
20        {
21            mid_wheel_linear( 0.120, -90, true )
22        }
23        else
24        {
25        }
26        if( ( MPSU_RmcLength >= 8 && MPSU_RmcData == 17 ) )
27        {
28            mid_wheel_linear( -0.120, 0, true )
29        }
30        else
31        {
32        }
33        if( ( MPSU_RmcLength >= 8 && MPSU_RmcData == 15 ) )
34        {
35            mid_wheel_linear( 0.000, 0, false )
36        }
37        else
38        {
39        }
40        if( ( MPSU_RmcLength >= 8 && MPSU_RmcData == 10 ) )
41        {
42            mid_wheel_angular( 30.0 )
43        }
44        else
45        {
46        }
47        if( ( MPSU_RmcLength >= 8 && MPSU_RmcData == 23 ) )
48        {
49            mid_wheel_angular( -30.0 )
50        }
51        else
52        {
53        }
54    }
55 }

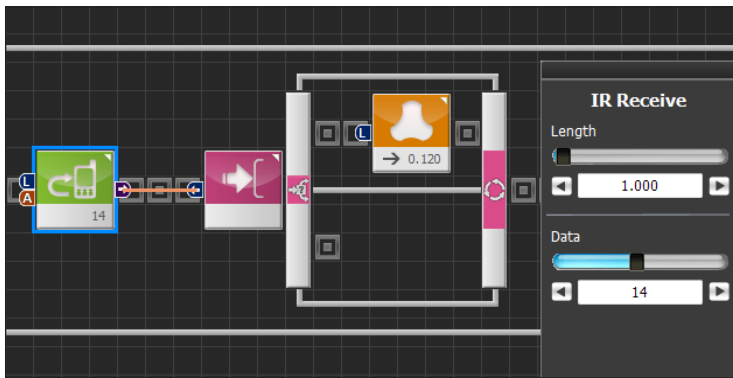
```

Step by Step



01 Loop

Select Loop module and dock to Start Point, Set Condition in property window to Forever since loop will repeat continuously to check for remote control signal.



02 Remote Control Button

Place IRReceive module and set property window values. Length in IRReceive property window refers to duration of the button press and Data refers to the number assigned to the remote control button.

- Length: 1.0 (Unit:Secod)
- Data : 14

IRReceive module outputs TRUE when remote control button representing 14 is pressed for more than 1s.

When TRUE, place the wheel modules so robot may mover forward.

- Mode: Linear
- Linear Velocity: 0,120
- Enable Offset Angle: False

Refer to the table for other buttons.

IRReceive Property	Wheel Property
Length: 1.0 Data: 13	Left sideway movement Mode: Linear Linear Velocity: 0,12 Enable Offset Angle: True Offset Angle: 90
Length: 1.0 Data: 16	Right sideway movement Mode: Linear Linear Velocity: 0,12 Enable Offset Angle: True Offset Angle: -90
Length: 1.0 Data: 17	Reverse Mode: Linear Linear Velocity: -0,12 Enable Offset Angle: False
Length: 1.0 Data: 15	Stop Mode: Linear Linear Velocity: 0,0 Enable Offset Angle: False
Length: 1.0 Data: 10	Left rotation Mode: Angular Angular Velocity: 30,0
Length: 1.0 Data: 23	Right rotation Mode: Angular Angular Velocity: -30,0



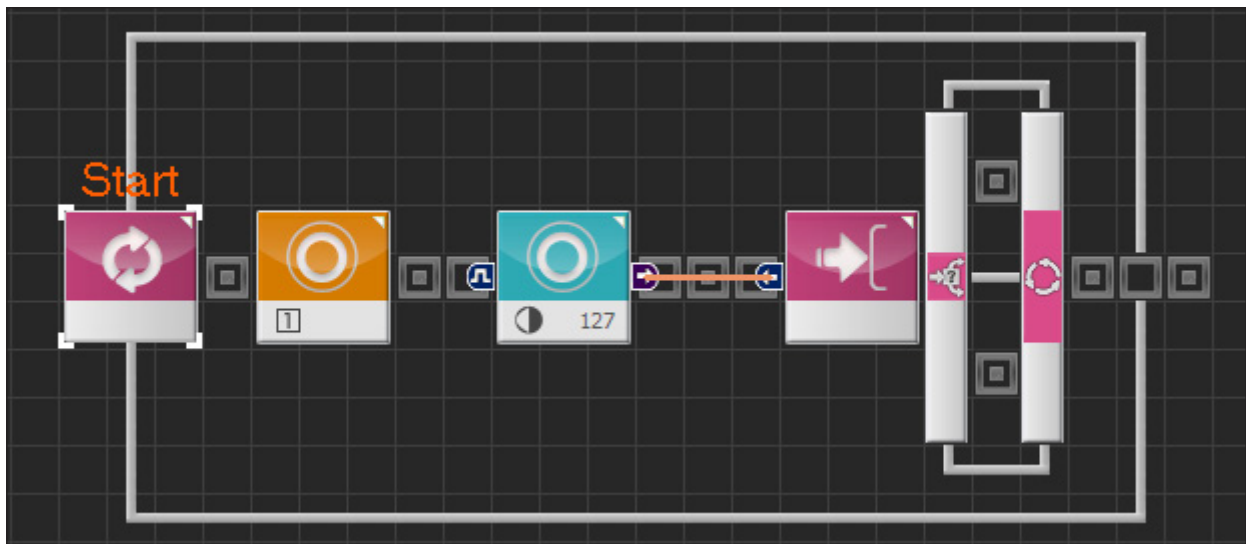
03 Compile, Download, Run

Click left compile button to compile the program. If there is no error, click right download button to download the program to MID. Click run button to run the program once download is complete

4.8 Vision Application

Just as human receives most sensory data input through the eyes, robot also receives most data through the vision camera. Hovis Genie uses the camera installed in the MID as its eyes. Variety of application programs can be created using DR-Visual Logic combined with vision processing functions such as Brightness, Motion detection, and Color detection.

Vision related Modules in DR-Visual Logic are divided between the vision capture modules and process modules. Vision Capture module is the capture module and process modules are Vision Sensor and Vision Result modules. Vision Sensor and Vision Result modules use the image captured by the Vision Capture module. Next two diagrams with Vision Capture, Vision Sensor, and Vision Result modules placed next to each other show how the image process flows in DR-Visual Logic (Loop module is required to run the image process procedure continuously. Loop module would not be required for single image process procedure.)



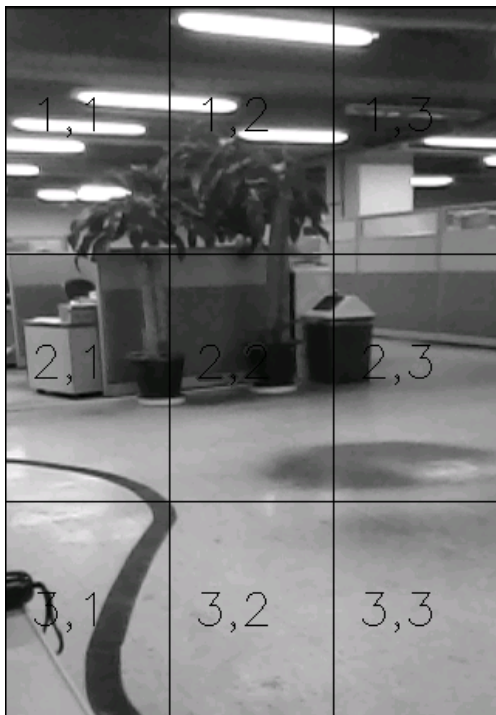
[Vision Capture – Vision Sensor]



[Vision Capture – Vision Result]

Vision Sensor and Vision Result modules where actual image processing takes place require Grid and Region to be set in the property window. Grid is used to distinguish specific areas in captured image by dividing the total image area by rows and columns into small rectangular areas. Region refers to the subdivided rectangular area (1,1) ~ (row, col).

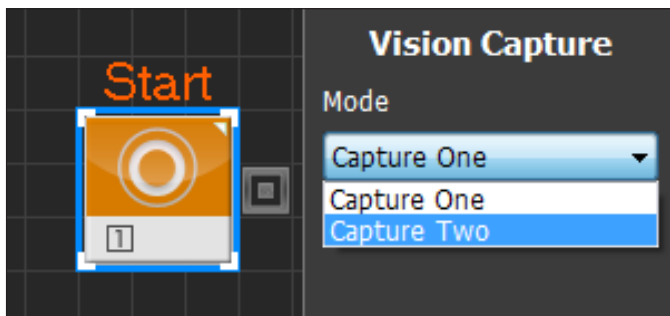
If captured image Grid Size (Row x Column) is set as 3 x 3, image will be divided into 9 areas as shown in the diagram below.



Region is the area on the grid where image processing is required. Set Region to (2,2) if the Grid Size is 3x3 and the area of interest is the center section.

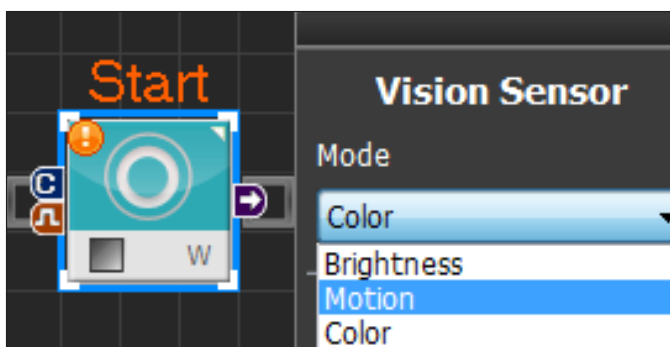
Diagram below shows the property window content of the image processing related modules in DR-Visual Logic.

Vision Capture Module



Mode > Capture One: Capture image once
 Mode > Capture Two: Capture image twice
 Interval: Interval between the first and 2nd image capture (1~1000ms)
 Used for motion detection

Vision Sensor and Vision Result Module



(1) Vision Sensor

Vision Sensor module processes image captured by Vision Capture module and outputs TRUE or FALSE.

Mode > Brightness

Output TRUE if average brightness of the image from the region is less than or equal to Threshold value
 Brightness Threshold : Critical brightness value (0~255)

Mode > Motion:

Output TRUE if movement in the region is greater than or equal to Threshold value. 2 captures required.
 Motion Threshold : Critical movement value (0~255)

Mode > Color

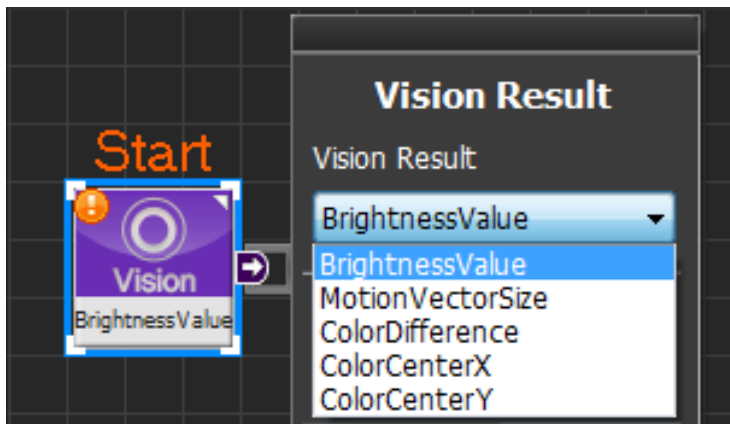
True if equal to or smaller than assigned color Threshold value.

Select from Color : White, Red, Green, Blue, Yellow, Black

Color Threshold : Assigned Color critical value (0~255)

(2) Vision Result

Vision Result module processes captured video by Vision Capture module and outputs applicable value.



Vision Result > BrightnessValue

Outputs average brightness of the region image.

Vision Result > MotionVectorSize

Outputs size of the movement within the region.

Vision Result > ColorDifference

Outputs the difference between the detected color in the region and the assigned color.

Select from Color : White, Red, Green, Blue, Yellow, Black

Vision Result > ColorCenterX

Vision Result > ColorCenterY

If object with assigned color is detected within the region, output x or y coordinate of the center of the object (Coordinate is based on the size of the MID screen 320 x 480)

Select from Color : White, Red, Green, Blue, Yellow, Black

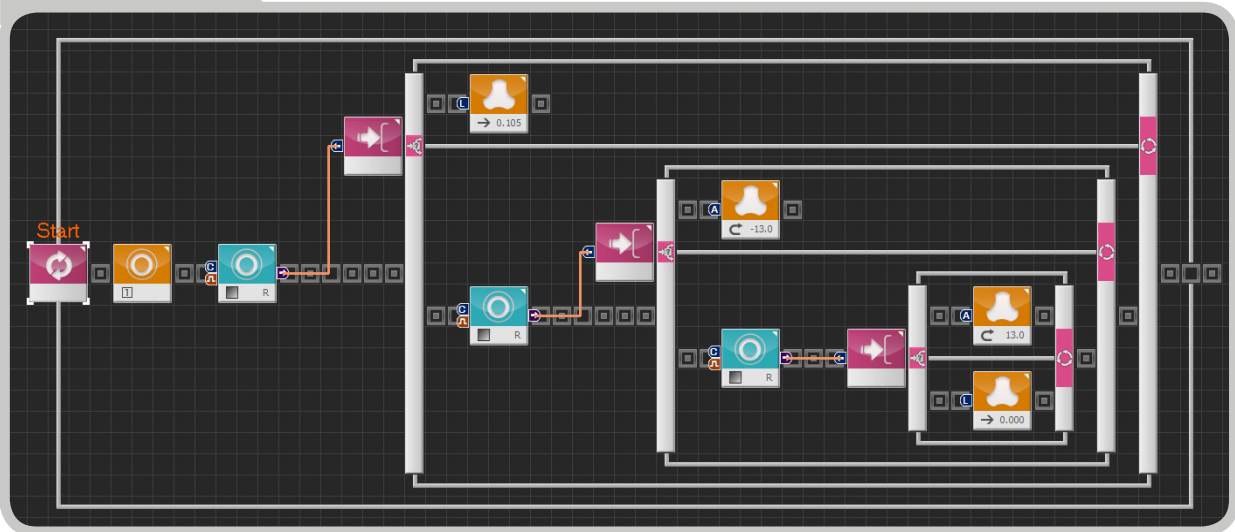
■ Example Description

In this example, robot detects and follows the red object. Captured MID camera image is divided into 3 x 3 Grid Size. Robot will move forward if red object is found in Region (2,2). If object is found in region (2,1) or (2,3), robot will turn right or left and follow the object.

Entire Program

1. Loop – Infinite Loop
2. Vision Capture – Video capture with MID camera
3. Vision Sensor – Move forward when red object detected in Grid Size (3X3) Region (2,2).
4. Vision Sensor – Rotate right when red object detected in Grid Size (3X3) Region (2,1)
5. Vision Sensor – Rotate left when red object detected in Grid Size (3X3) Region (2,3)

Graphic



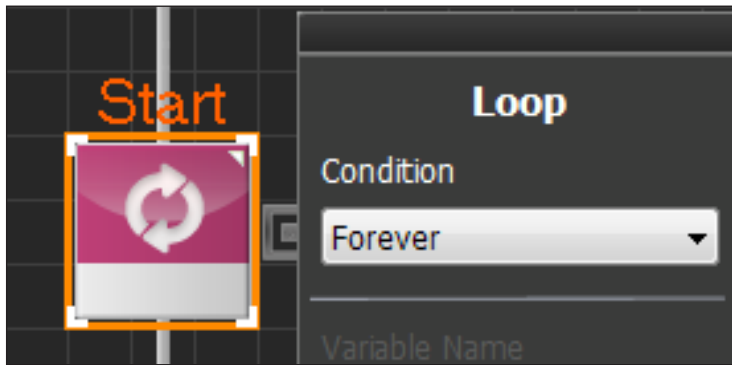
C-Like

```

1 void main()
2 {
3     while( true )
4     {
5         mid_vision_capture_one()
6         if( ( VISION_ColorDifference[ 3, 3, 2, 1 ] <= 30 ) )
7         {
8             mid_wheel_linear( 0.105, 0, false )
9         }
10        else
11        {
12            if( ( VISION_ColorDifference[ 3, 3, 2, 1, 1 ] <= 30 ) )
13            {
14                mid_wheel_angular( -13.0 )
15            }
16            else
17            {
18                if( ( VISION_ColorDifference[ 3, 3, 2, 3, 1 ] <= 30 ) )
19                {
20                    mid_wheel_angular( 13.0 )
21                }
22                else
23                {
24                    mid_wheel_linear( 0.000, 0, false )
25                }
26            }
27        }
28    }
29 }

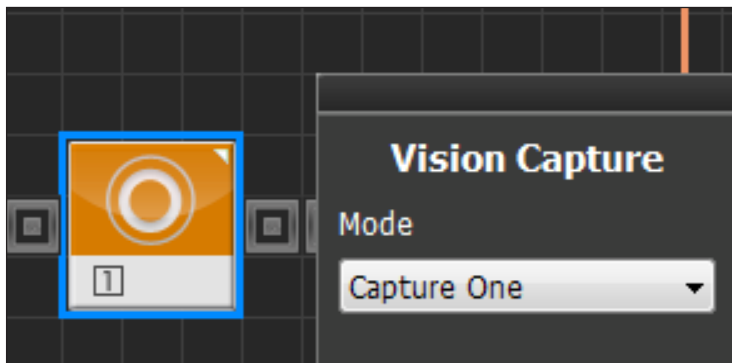
```

Step by Step



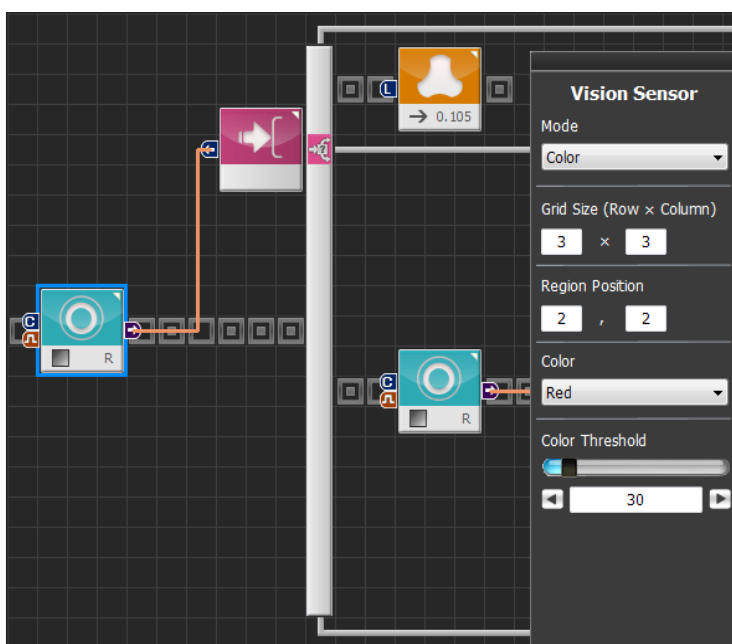
01 Loop

Place Loop module with Forever condition to continuously process the image,



02 Vision Capture

Capturing image. Set Mode to Capture One to capture single photo.

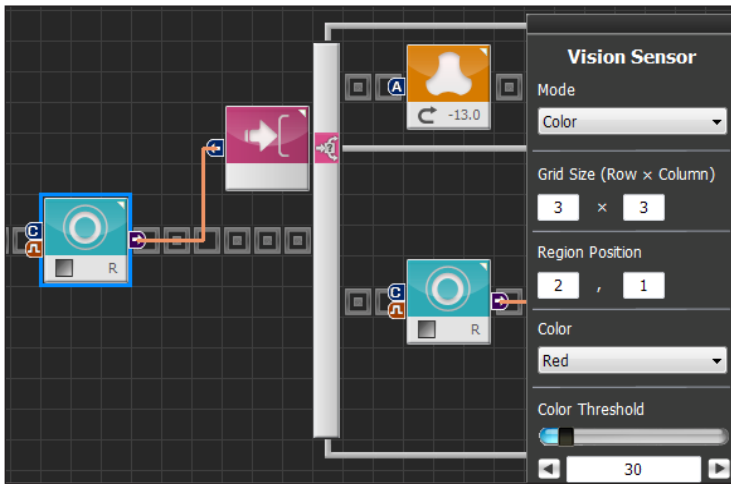


03 Vision Sensor (Robot forward movement condition)

Place Vision Sensor module. If red object is detected in the center, move robot forward, otherwise place another condition. Set Vision Sensor module properties as follows.

- Mode: Color
- Grid Size (Row x Column): 3 x 3
- Region Position: 2, 2
- Color: Red
- Color Threshold: 30

※Vision Sensor will recognize an object as being red if the color variation is within 30. Increase Color Threshold value to increase the tolerance margin in recognizing object as red.

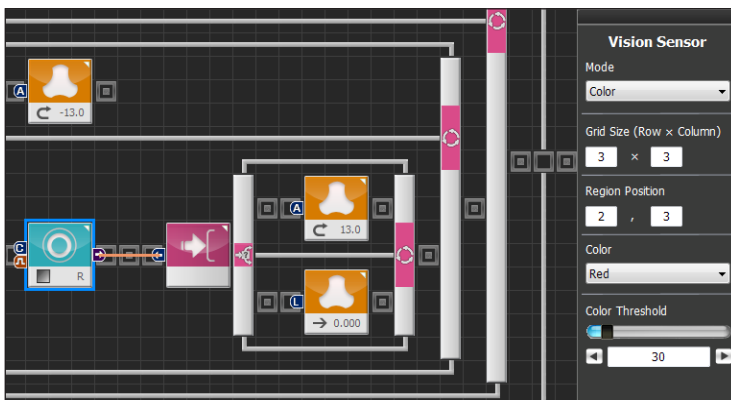


04 Vision Sensor (Right rotation condition)

When Vision Sensor module output value in section 03 is FALSE, right side will be checked by Vision Sensor module for red object. Robot will rotate right if red object is detected.

Set Vision Sensor module properties as follows.

- Mode: Color
- Grid Size (Row x Column): 3 x 3
- Region Position: 2, 1
- Color: Red
- Color Threshold: 30



05 Vision Sensor (Left rotation and stop)

When Vision Sensor module output value in section 04 is FALSE, left side will be checked by Vision Sensor module for red object. Robot will rotate left if red object is detected.

Set Vision Sensor module properties as follows.

- Mode: Color
- Grid Size (Row x Column): 3 x 3
- Region Position: 2, 3
- Color: Red
- Color Threshold: 30



06 Compile, Download, Run

Click left compile button to compile the program. If there is no error, click right download button to download the program to MID. Click run button to run the program once download is complete