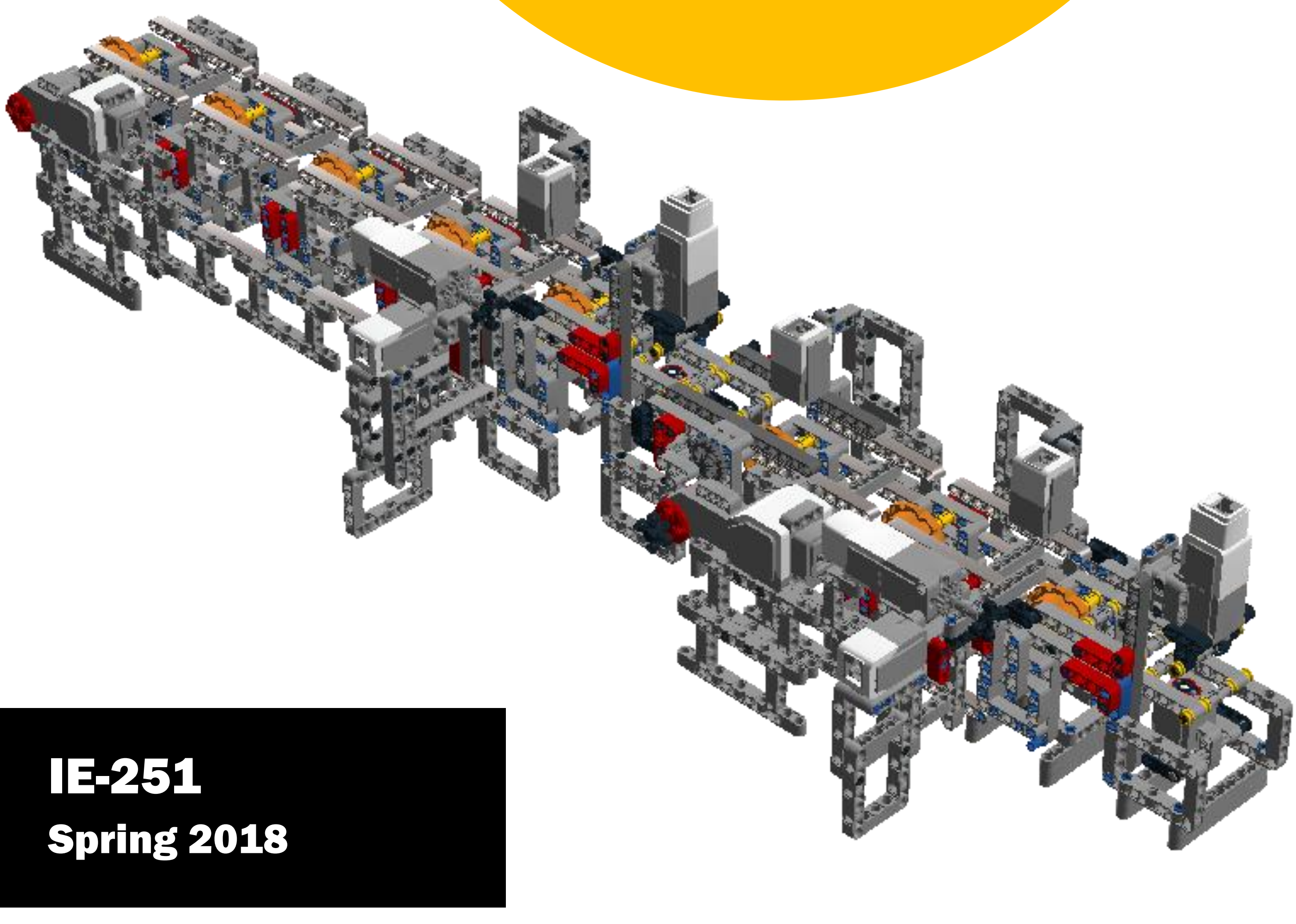
LEGO Production Line

Instruction



IE-251

Spring 2018

LEGO Production Line
KAIST

This instruction is intended for the laboratory activities of IE-251 Manufacturing Process Innovation at Korea Advanced Institute of Science and Technology.

LEGO Robotics Introduction

LEGO Mindstorms supports robotic function in the forms of motors and sensors that enables the creation of a functioning automatic production line as hands-on laboratory activities.

LEGO Production Line

A complete production line is assembled by combining the components of the LEGO EV3 component. You will learn how to build the production line from zeros. A detail elements-by-elements are provided in a separate folder.

MATLAB Programming

MATLAB programming is used to run the LEGO production line. You will learn how to run, program , and improve the LEGO production line. Scripts are provided in a separate folder.

Class Project/Assignment

01. LEGO Robotics Introduction

A. LEGO Mindstorms

LEGO Mindstorms supports robotic function in the forms of motors and sensors that enables the creation of a functioning automatic production line as hands-on laboratory activities. A core set of LEGO EV3 consists of robotic component and static-construction LEGO brick.

Figure 1 LEGO Mindstorms Core Set Component

a. Robotic components

- EV3 **Brick** is the main controller in LEGO Robotics. One brick can accommodate four sensors and four motors simultaneously. Connected through USB, Wi-Fi, or Bluetooth, LabVIEW or MATLAB programming can program EV3 Brick.

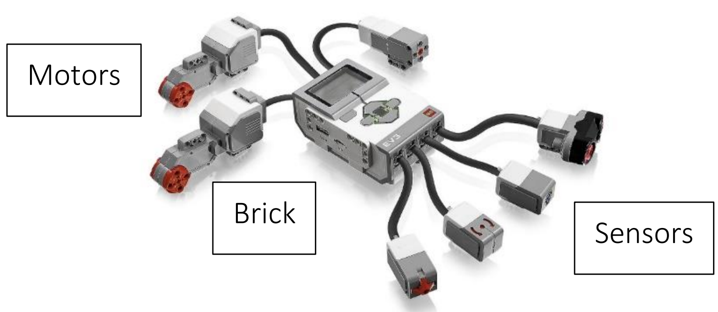


Figure 2 LEGO EV3 Robotic Component

- LEGO EV3 has two types of **motors**, large with a capacity of 160-170 RPM and medium with a capacity of 240-250 RPM.
- To control the construction, LEGO EV3 employs light **sensors**, touch sensors, gyro sensors and infrared sensors.

b. Static construction components
Static **blocks** consist of beam, rotating parts, wheels, bolt, gears, and other structure.

02. LEGO Production Line Construction

A. Production Line Model

A complete production line is assembled by combining the components of the LEGO EV3 component.

The model of Lego Production Line follows the two-machine-one-buffer system [ref1].

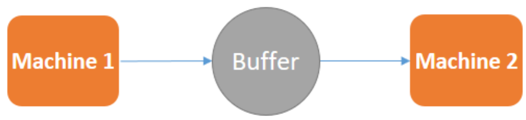


Figure 3 Two Machine One Buffer Model

Raw material enters the system from machine 1, buffer, machine 1, and finally exit the system after completing the process. Each machine has specific parameters of failure, repair, and processing rate.

Starvation in machine 2 happens when machine 1 fails and the buffer is empty. Blockage happens when machine 2 fails, machine 1 is working, and the buffer becomes full. Machine 1 is assumed to never starve and machine 2 never blocked.

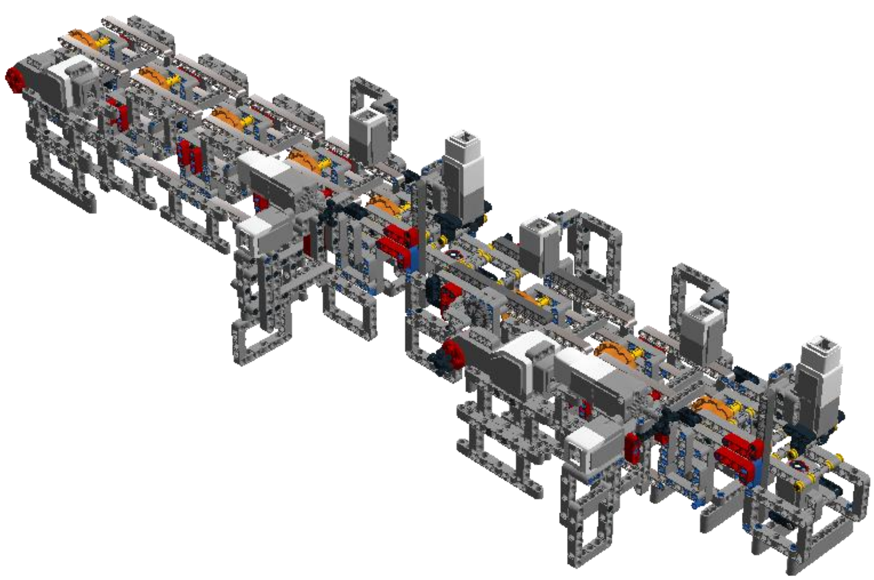


Figure 4 LEGO Production Line (LPL)

B. Lego Production Line (LPL) Construction

LPL construction consists of two machines and automatic material handling system (AMHS) with conveyor as the transport system. LPL adds feeder conveyor before machine 1 to the construction to complete the AMHS and the assumption of starvation.

Figure 5 Two Machine One Buffer with Feeder Conveyor

With this configuration, a material flows to the system through feeder conveyor to machine 1, to buffer conveyor, to machine 2, and exit the system. Materials enter the feeder conveyor and exit from machine 2 are handled manually.

a) AMHS' conveyor

The AMHS' conveyor is a transport system that delivers materials from one point to another point. In LPL, there are two conveyor systems with their specific function. Feeder conveyor transports materials to the first machine while buffer conveyor transports materials from machine 1 to machine 2.

Except for the length, both conveyors have similar construction. It has two main components, belt track and the framework. The conveyor is assembled by attaching LEGO track elements to form a belt shape. The framework consists of several frame submodule with beams and sprocket to rotate the conveyor belt.

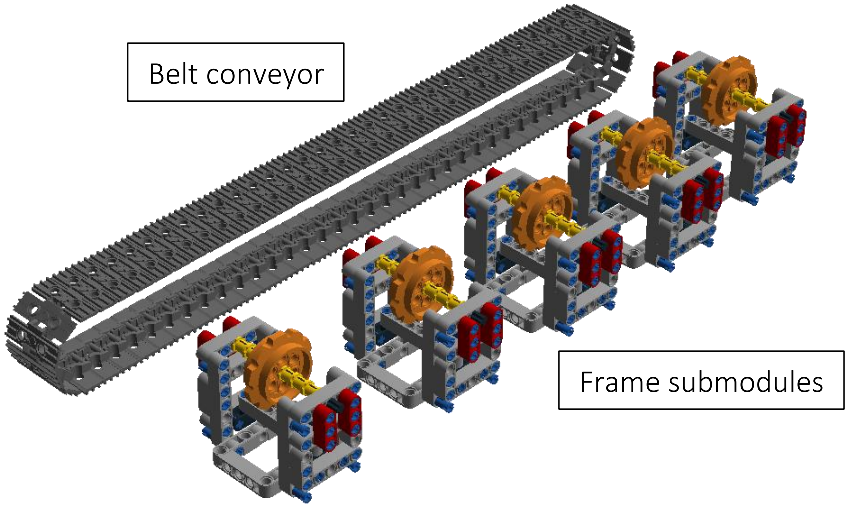


Figure 6 Conveyor Sub-modules

A large motor is connected to the first axle of the frame to rotate the belt. Also, To level the motors and belt, a foundation construction is built by combining several beams.

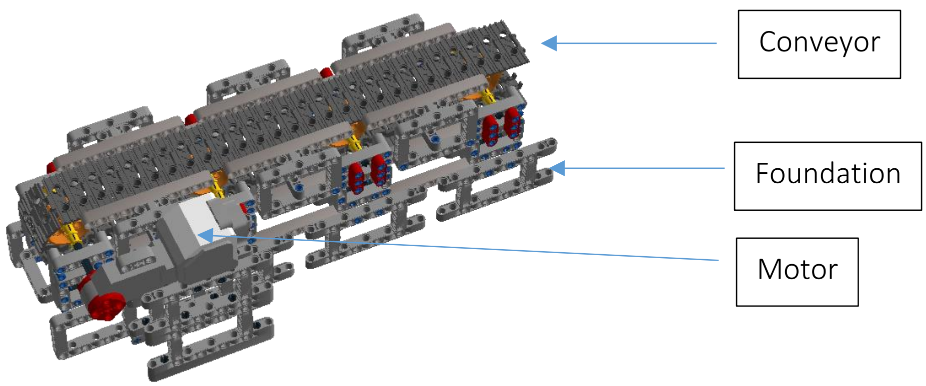


Figure 7 Conveyor and Motor Configuration

b) Gate mechanism

To control the flow of material from a conveyor to machine, LPL employs gate mechanism. Main program controls the opening and closing of the gate by coordinating touch sensor and motor. The gate is formed by a cam construction using gears and beam. It transfers rotary motion from the gear into linear motion of the beam gate. After the gate opens and let one material go through, the main program will command the motor to position the beam back into closed position, keeping the rest of the queued materials behind the gate.

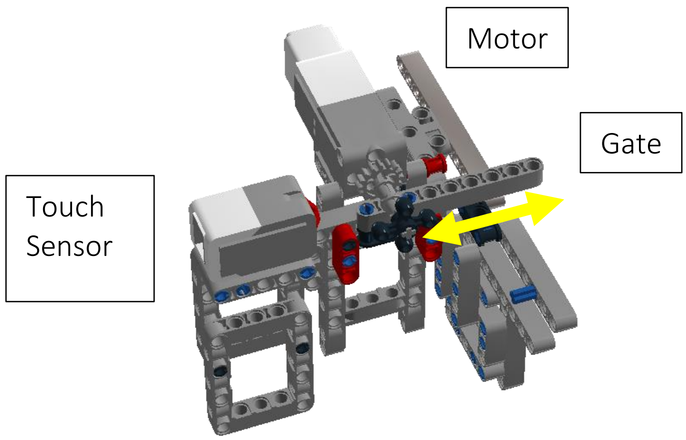


Figure 8 Gate Module

c) Machine

LPL machine is built from static components of LEGO and one light/color sensors. The machine mimics a process by holding a part in *t* seconds. Light sensors detect a material by recognizing the light intensity transmitted back to the sensors. When it recognizes a part, machine will pause the time for *t* seconds.After a process, the part is transported out of the machine by a motor and v-belt configuration, attached to the machine.

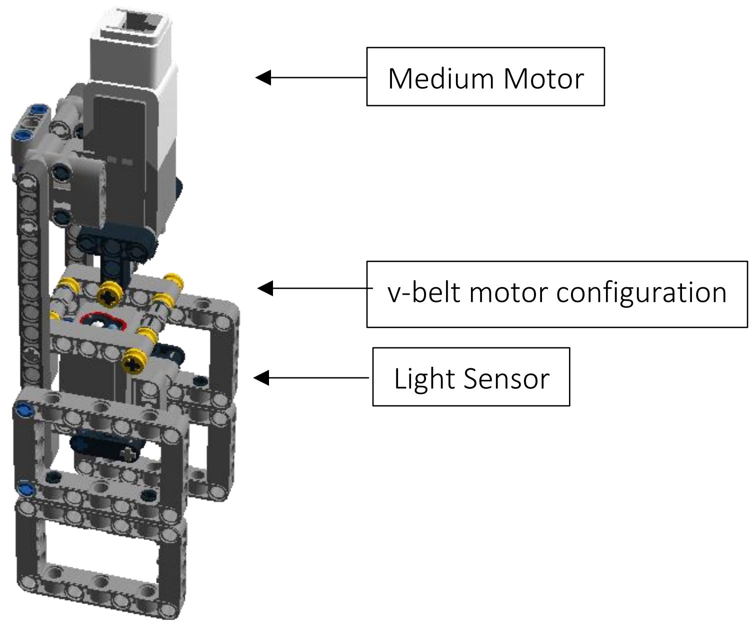


Figure 9 Machine Module

d) Control Integration

In the operational practice of production line, interaction between machine and buffer constitutes a more complicated control system that requires real-time coordination. Sensors are installed to manage communication between main program, conveyors and machines.

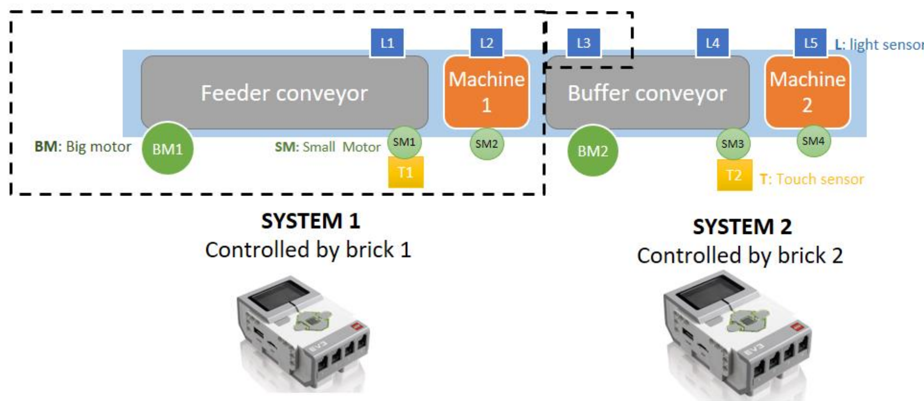


Figure 10 LEGO Production Line Control Configuration

The mechanism of the LEGO production line is as follows.

01. Light sensors L1, L2, L3, touch sensors T1, big motor BM1, and small motor SM1 are controlled by brick 1, and the rest are by brick 2.
02. Main program runs motor BM1 and BM2 to activate feeder conveyor and buffer conveyor.
03. Raw material is placed manually to the feeder conveyor and enters the system.
- Light sensors L1 above the feeder conveyor notifies the incoming material.
04. When machine 1 is idle and working (not broken), the gate allows the first material in the queue to enter machine 1.
- Events of failure and repair is generated in the main program.
- Light sensors L2 identifies whether machine 1 is idle or not.
05. When machine 1 is broken, the gate blocks the inlet to machine 1.
- Motor SM1 operates the gate with information about gate position from touch sensors T1.
- When the buffer conveyor is full, sensor light L3 above the feeder conveyor blocks the movement of the material.
06. Machine 1 mimics a process step by holding the material for *t* second.
07. If in the middle of the process, the machine breaks, the material stay in the machine until it repaired.

- Once the step success, the medium motors SM1 on top of the machine will trigger the attached v-belt to move the material to the buffer conveyor.

08. If machine 2 idle and is not broken, the second gate will let the first material in the buffer conveyor to enter machine 2.

- The existence of material is signaled by touch sensors L4
- Motor SM2 operates the gate with information about gate position from touch sensors T2.
- Machine 2 has the same failure mechanism with machine 1.

09. Once the process in machine 2 is finished, v-belt will transport the material to exit the system.

- Material information is signaled by touch sensors L5

C. Building Instruction

There are two options to open detail element-by-element guide of LPL

1. Pdf files on folder PDF Guide.
2. For an interactive guide, install LEGO digital designer, and open file with extension .lxf on folder LDD Guide.

LEGO Digital Designer is a design software that allows user to virtually design a construction through virtual LEGO bricks. Interactive and element-by-element instruction of LPL is available through LEGO Digital Designer. The software is accessible through https://www.lego.com/en-us/ldd.

After opening the file, click *Building Guide Mode* or F7 to generate the interactive instruction.

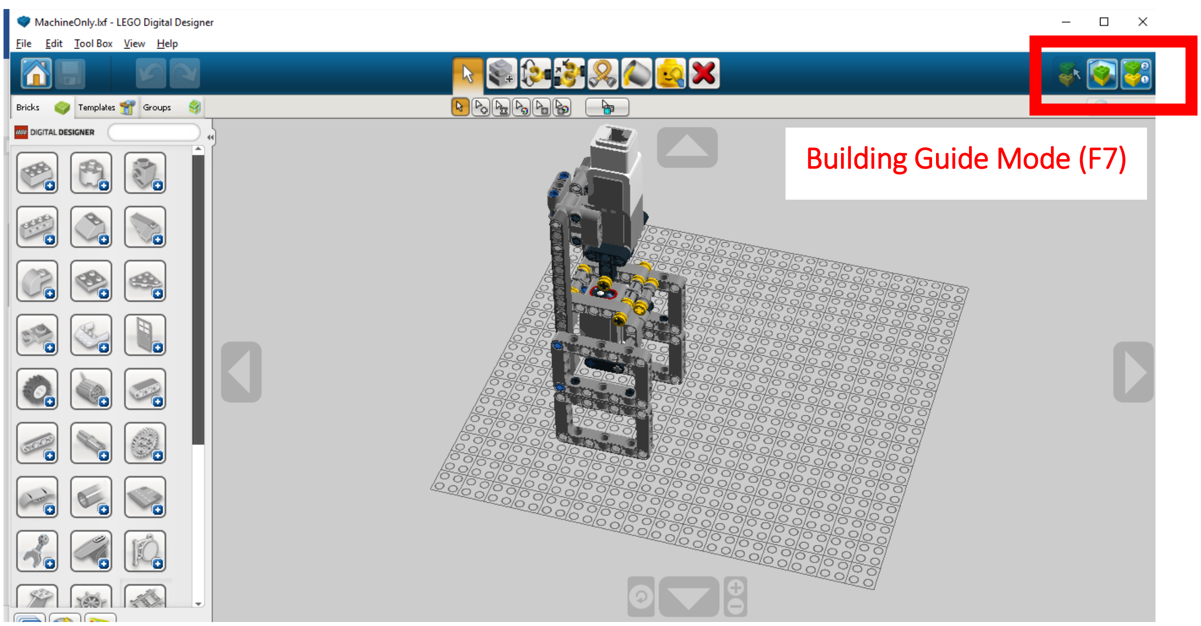


Figure 11 LEGO Digital Designer

Table 1 Building Guide Source

Modules	Pdf guide	LDD guide
Machine	MachineOnlyGuide.pdf MachineGuide.pdf	MachineOnly.lxf Machine.lxf
Feeder conveyor	ConveyorBasicGuide.pdf LongConveyorGuide.pdf	ConveyorBasic.lxf LongConveyor.lxf
Gate cam	CamGuide.pdf	Cam.lxf
Whole LPL	All.pdf	All.lxf

03. LEGO Production Line MATLAB Programming

A. MATLAB-EV3 Support Tutorial

a. MATLAB basic

KAIST provide license and software file from https://kftp.kaist.ac.kr/index.php.

For basic knowledge about MATLAB please refer to the references below

- Data Structure lecture note
- Online tutorial https://www.tutorialspoint.com/matlab/index.htm
  For beginner user, you are advised to start with MATLAB tutorial from overview tab until array tab. It will cost you around 3-4 hours to complete the tutorial
- Mathworks website https://kr.mathworks.com/

b. MATLAB LEGO basic programming

MATLAB programming is used to run the LEGO production line. The syntax are simple and easy to use for novice programmer. The following are MATLAB program for LEGO to computer connection.

- Connection with USB
  lego1 = legoev3('usb')

- Connection with Wi-Fi (Additional Wi-Fi dongle required)
  lego1 = legoev3('wifi','brickIP address','brick_product_ID')

- Connection with Bluetooth
  lego1 = legoev3('bt',' ,'brick_product_ID')

- Termination of LEGO connection
  clear lego1

For more basic tutorial such as reading sensor and motor value, visit https://www.mathworks.com/hardware-support/lego-mindstorms-ev3-matlab.html.

B. LEGO Production Line MATLAB Configuration

Each LEGO production line has 5 sensors, 2 touch sensors, 6 motors. Therefore, to accommodate all of the components, we require 2 bricks. MATLAB can only recognize 1 brick, thus we need to open two MATLAB programs, and therefore two different sets of MATLAB script, ***System 1*** and ***System 2***. The scripts are available on folder MATLAB files.The possibility of connection configuration are as follows:

a. Two computers with two USB connection brick.
b. One computer with one USB connection brick and one Wi-Fi connection brick.

Figure 12 LEGO-MATLAB Connection

There are six m-files in each folder. Each m-files in the following section are accompanied by flowchart describing the flow and sequence of LEGO Production Line operation.

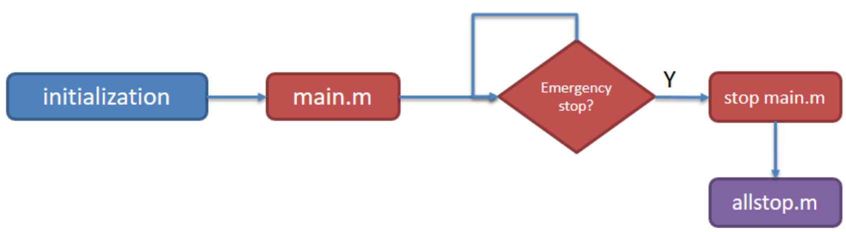


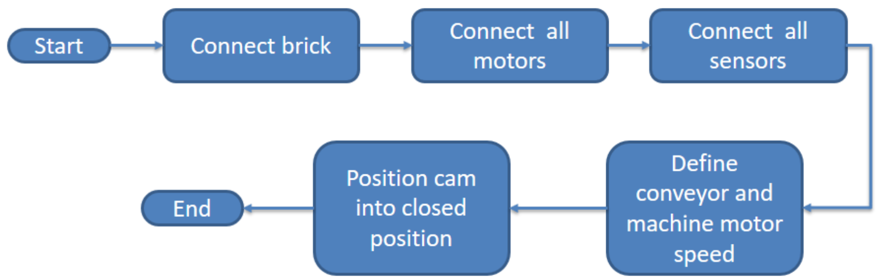
Figure 13 Flow of MATLAB Program

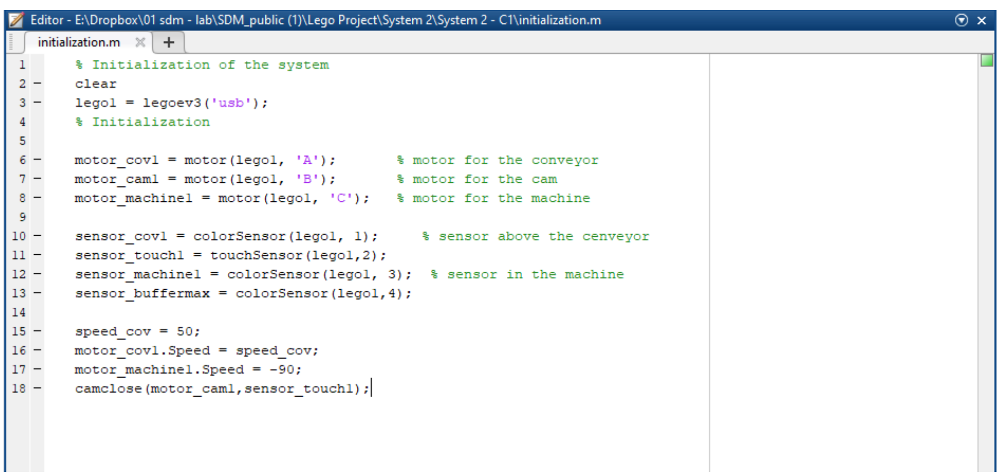
a) Initialization

Upon starting the program, initialization is needed to define all the robotics component into the program. Initialization for both system follows the definition in Table 2.
File name: initialization.m

Table 2 Brick Port Configuration

Brick 1 – System 1		Brick 2 – System 2	
BM1	Port A	BM2	Port A
SM1	Port B	SM3	Port B
SM2	Port C	SM4	Port C
L1	Port 1	L4	Port 1
L2	Port 2	L5	Port 2
L3	Port 3	T2	Port 3
T1	Port 4		



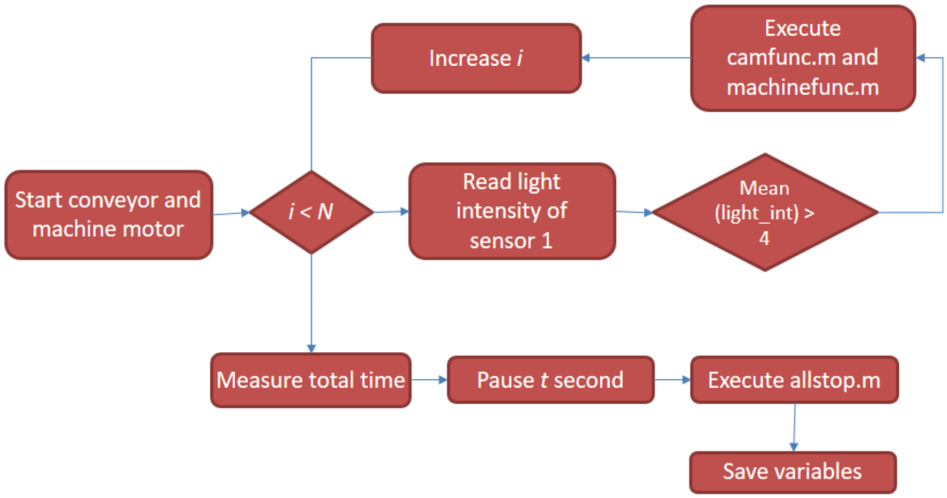



```
% Initialization of the system
clear
lego1 = legoev3('usb');
% Initialization

motor_cov1 = motor(lego1, 'A');       % motor for the conveyor
motor_cam1 = motor(lego1, 'B');       % motor for the cam
motor_machine1 = motor(lego1, 'C');   % motor for the machine

sensor_cov1 = colorSensor(lego1, 1);      % sensor above the cenveyor
sensor_touch1 = touchSensor(lego1,2);
sensor_machine1 = colorSensor(lego1, 3);  % sensor in the machine
sensor_buffermax = colorSensor(lego1,4);

speed_cov = 50;
motor_cov1.Speed = speed_cov;
motor_machine1.Speed = -90;
camclose(motor_cam1,sensor_touch1);
```

Figure 14 Initialization

b) Main Program

After initialization, define *process time, failure rate, repair rate, repair time*, and execute main program to activate LEGO Production Line program. At this phase, material is placed manually into the feeder conveyor.
File Name: main.m



Editor - E:\Dropbox\01 sdm - lab\SDM_public (1)\Lego Project\System 2\System 2 - C1\main.m

initialization.m | main.m | allstop.m

```
allstop
global process_time; global repair_time; global failure_rate; global repair_rate; global n;global state;
state = cell(1,2); n = 1;
process_time = 3;
repair_time = 3;
failure_rate = 0.2; repair_rate = 0.9;
N = 100; m = 100;
i = 0;
light_int = zeros(1,m);
start(motor_cov1); start(motor_machine1);
tic;
while i < N %fq
    %%%%update light intensity information on coveyor color sensor.
    for j = 1 : m-1
        light_int(j) = light_int(j+1);
    end
    light_int(m) = readLightIntensity(sensor_cov1,'reflected');

    if mean(light_int) > 4     % condition for detection on conveyor
        camfunc(motor_cam1,sensor_touch1); % cam open, chip passes, cam close
        light_int = zeros(1,m); % reset light intensity info.
        machinefunc(lego1,motor_machine1,sensor_machine1,sensor_buffermax); % machine processing
        i = i + 1
    end
end
toc;
pause(2)
allstop;
save state1.mat state;
disp(length(state)-N)
```

Figure 15 Main Program

c) *Emergency stop*

The script is intended to stop all motors activities when the production line has to be stopped because of any failure.
File name: allstop.m

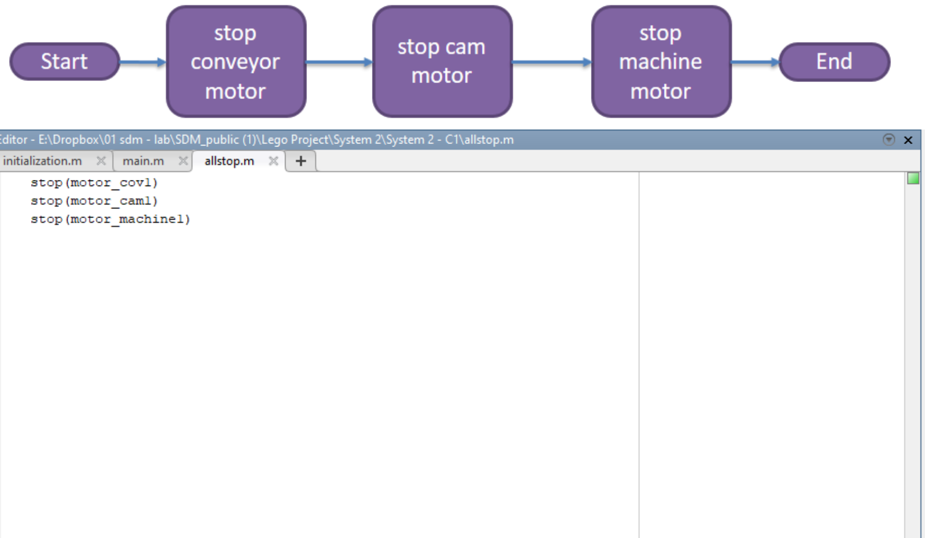


Figure 16 Emergency Stop