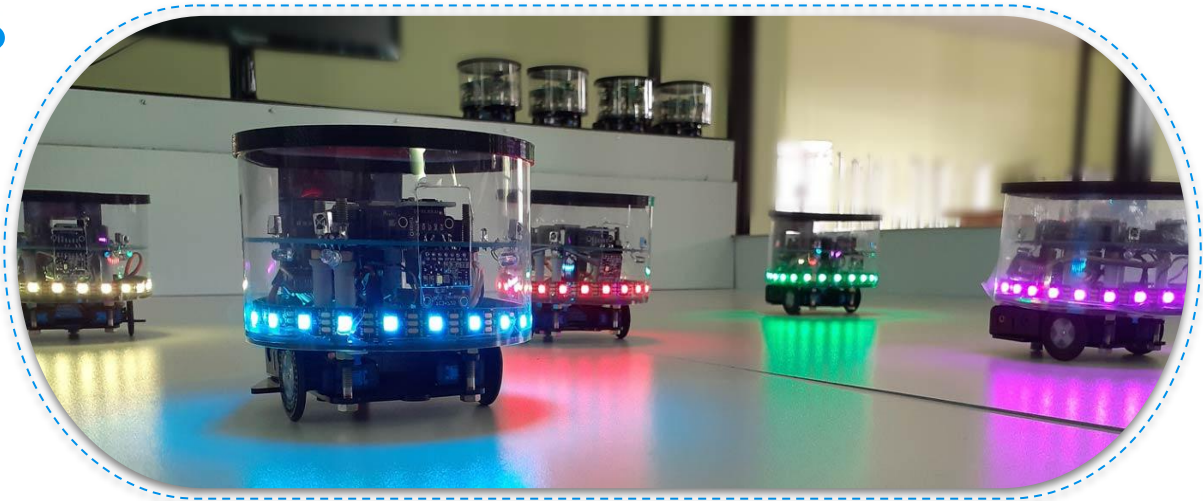


Swarm Intelligence Programming Framework

By Group 03



Members



Ekanayake S.M.

E/16/094



Madushanka H.M.K.

E/16/221



Perera A.L.H.E.

E/16/275

Supervisors



Prof. Roshan Ragel



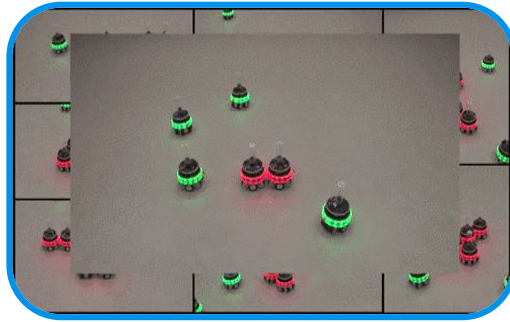
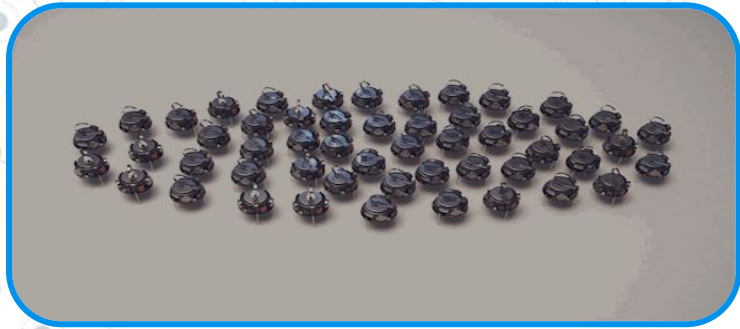
Dr. Isuru Nawinne



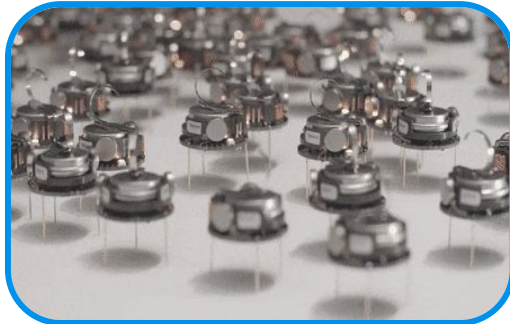
**Dr. Mahanama
Wickramasinghe**




Dr. Sithumini Ekanayake



Swarm Robotics



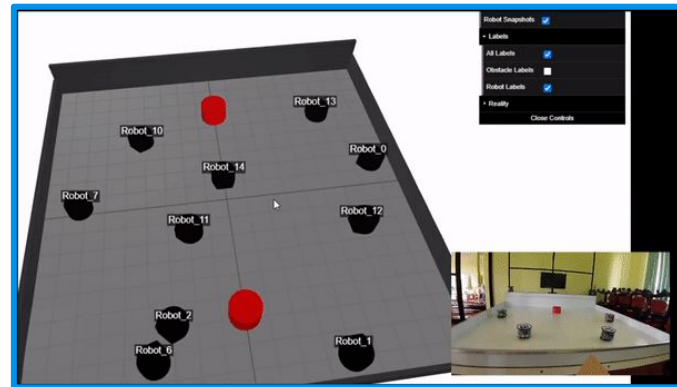
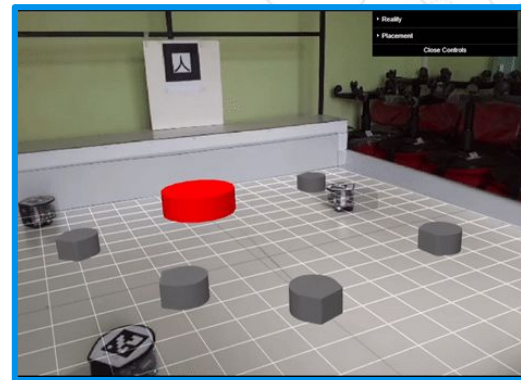
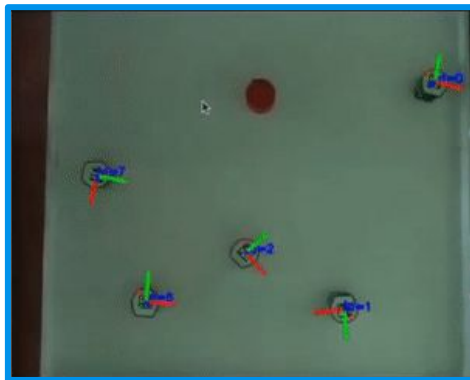
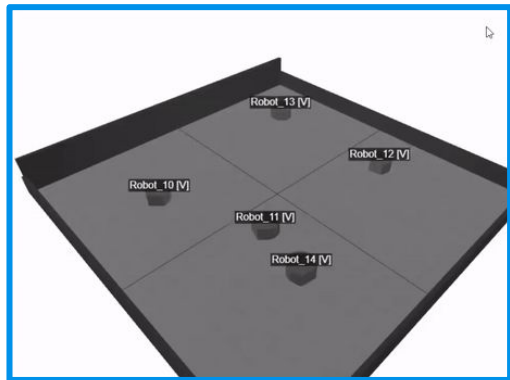


1

Introduction

Introduction to PeraSwarm Project

PeraSwarm





2

Problems

Issues that are going to solve

Problems

- ☹️ No framework that supports different physical and virtual robots.
- ☹️ Inability to programme multiple robots over the air.
- ☹️ Swarm programmers must be experts in programming.



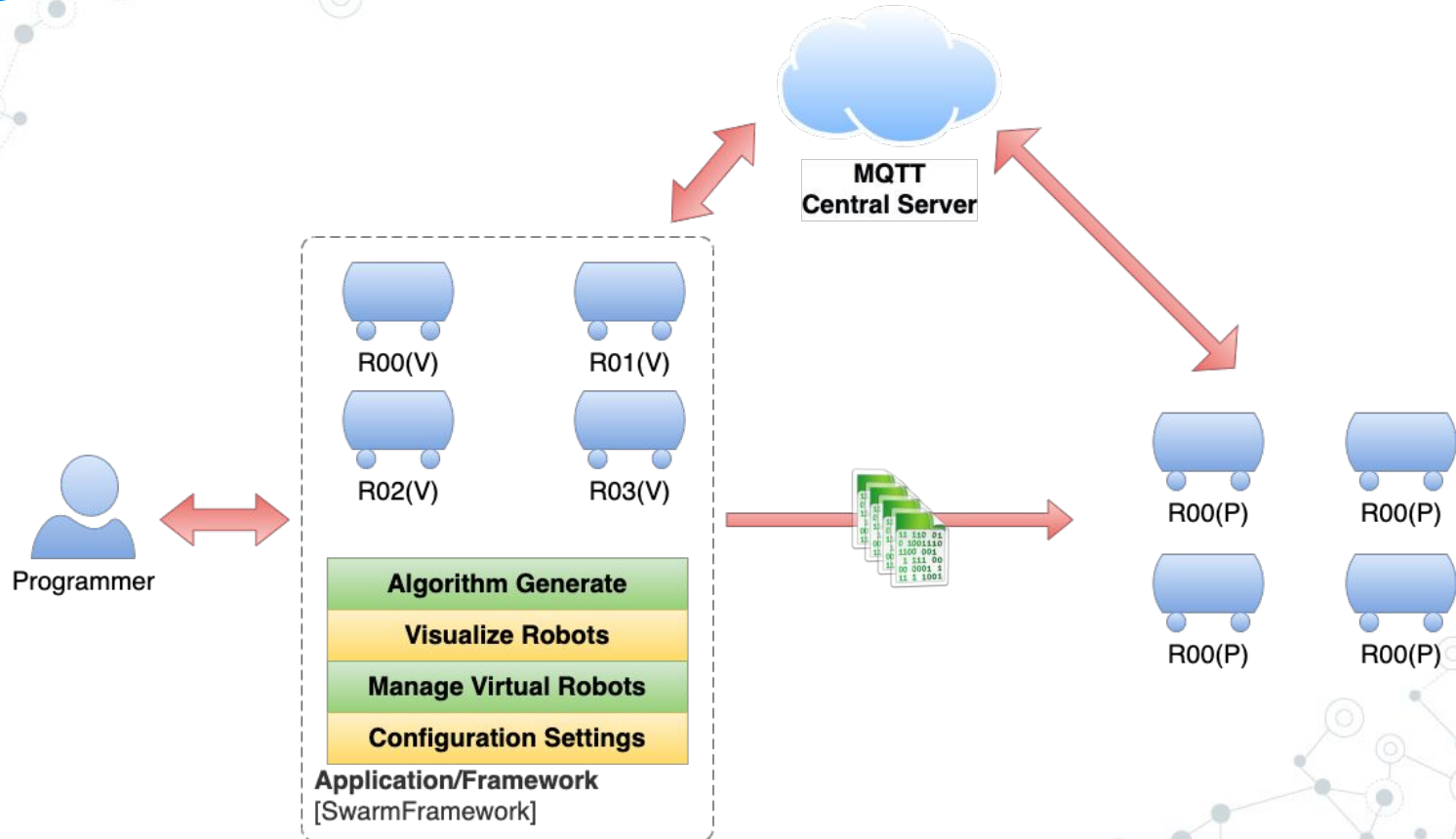
A decorative network diagram in the top-left corner, consisting of various sized grey circles connected by thin grey lines, forming a complex web-like structure.

3

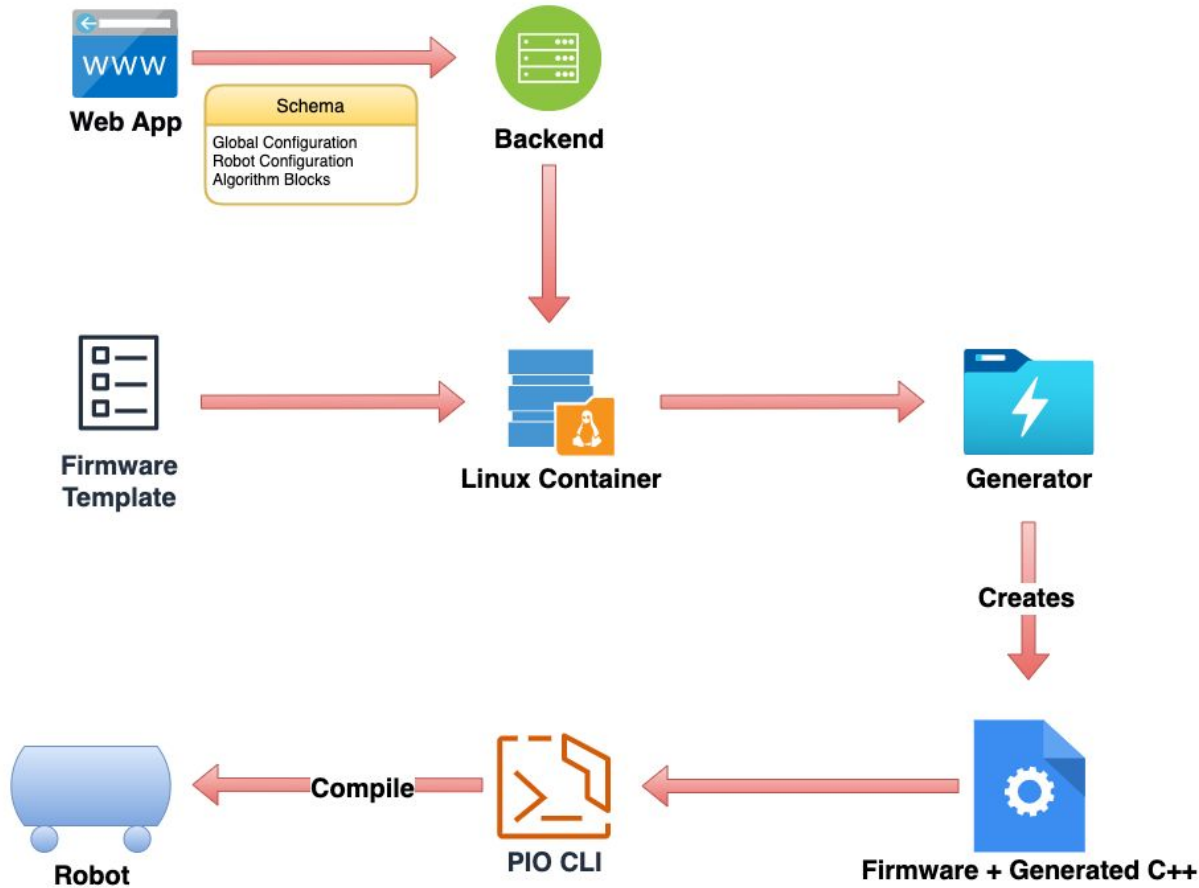
Solution

Solution Architecture

High level overview



Software Architecture



Roadmap

Remote Cross
Compiler
Development

1

Algorithms
Development &
Testing

3

Aggregator FSM
Testing

5

2

Code Generation &
Uploading to
Robots

4

Finite State
Machines (FSMs)
Testing

6

Testing SAR(Search
& Rescue on Arena)

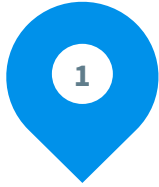


7

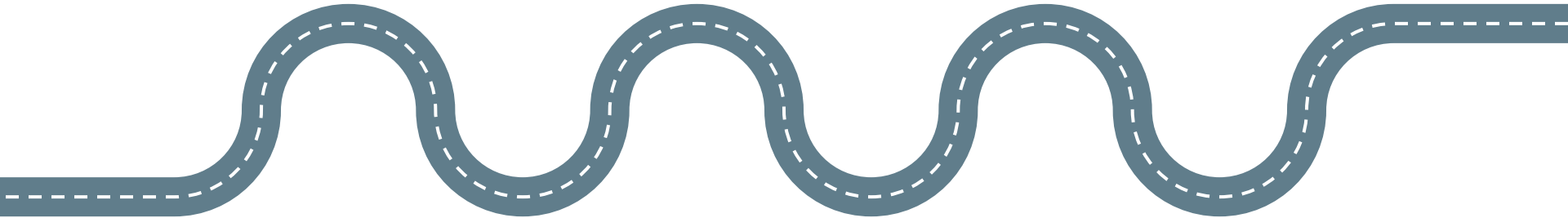
Progress

Current Progress Status

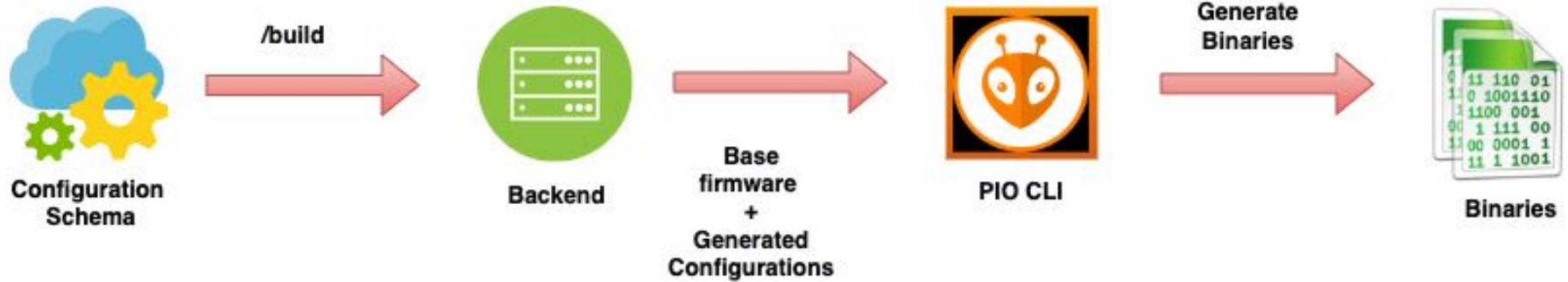
Roadmap



**Remote Cross Compiler
Development**



Remote Cross Compiler Development



Remote Cross



Configuration
Schema

/build

```
{
  features: [
    { name: 'ALGORITHM', value: 'ALGO_COLOR_TEST', isEnabled: true },
    {
      name: 'ENABLE_SERIAL_COMMUNICATION',
      value: 'ENABLE_SERIAL_COMMUNICATION',
      isEnabled: true,
      extra: [Array]
    },
    {
      name: 'NEOPIXEL_INDICATIONS',
      value: 'NEOPIXEL_INDICATIONS',
      isEnabled: true
    },
    { name: 'ENABLE_MEMORY', value: 'ENABLE_MEMORY', isEnabled: true },
    {
      name: 'ENABLE_MOTORS',
      value: 'ENABLE_MOTORS',
      isEnabled: false,
      dependencies: [Array]
    },
    {
      name: 'ENABLE_DISTANCE_SENSOR',
      value: 'ENABLE_DISTANCE_SENSOR',
      isEnabled: false,
      dependencies: [Array]
    },
    {
      name: 'ENABLE_NEOPIXEL_RING',
      value: 'ENABLE_NEOPIXEL_RING',
      isEnabled: false
    },
    {
      name: 'ENABLE_COLOR_SENSOR',
      value: 'ENABLE_COLOR_SENSOR',
      isEnabled: false
    },
    {
      name: 'ENABLE_COMPASS_SENSOR',
      value: 'ENABLE_COMPASS_SENSOR',
      isEnabled: false
    },
    {
      name: 'ENABLE_OTA_UPLOAD',
      value: 'ENABLE_OTA_UPLOAD',
      isEnabled: false
    },
    { name: 'ENABLE_MQTT', value: 'ENABLE_MQTT', isEnabled: true },
    { name: 'ENABLE_WIFI', value: 'ENABLE_WIFI', isEnabled: true }
  ]
}
```

Generate
Binaries



Binaries

Compiler Development

```
#pragma once
/*
  This is an auto-generated file.
 */

#define ALGO_COLOR_TEST

#define ENABLE_SERIAL_COMMUNICATION1

#define NEOPIXEL_INDICATIONS

#define ENABLE_MEMORY

#define ENABLE_MOTORS

#ifdef ENABLE_MOTORS
#define DRIVE_PWM
#define DRIVE_SERVO
#endif

#define ENABLE_DISTANCE_SENSOR

#ifdef ENABLE_DISTANCE_SENSOR
#define DISTANCE_GP2Y0A21YK0F
#endif

#define ENABLE_NEOPIXEL_RING

#define ENABLE_MQTT

#define ENABLE_WIFI

/* ----- End of file ----- */
```

**Base
Firmware
+
Generated
Configurations**



PIO CLI

Generate
Binaries



Binaries

Roadmap

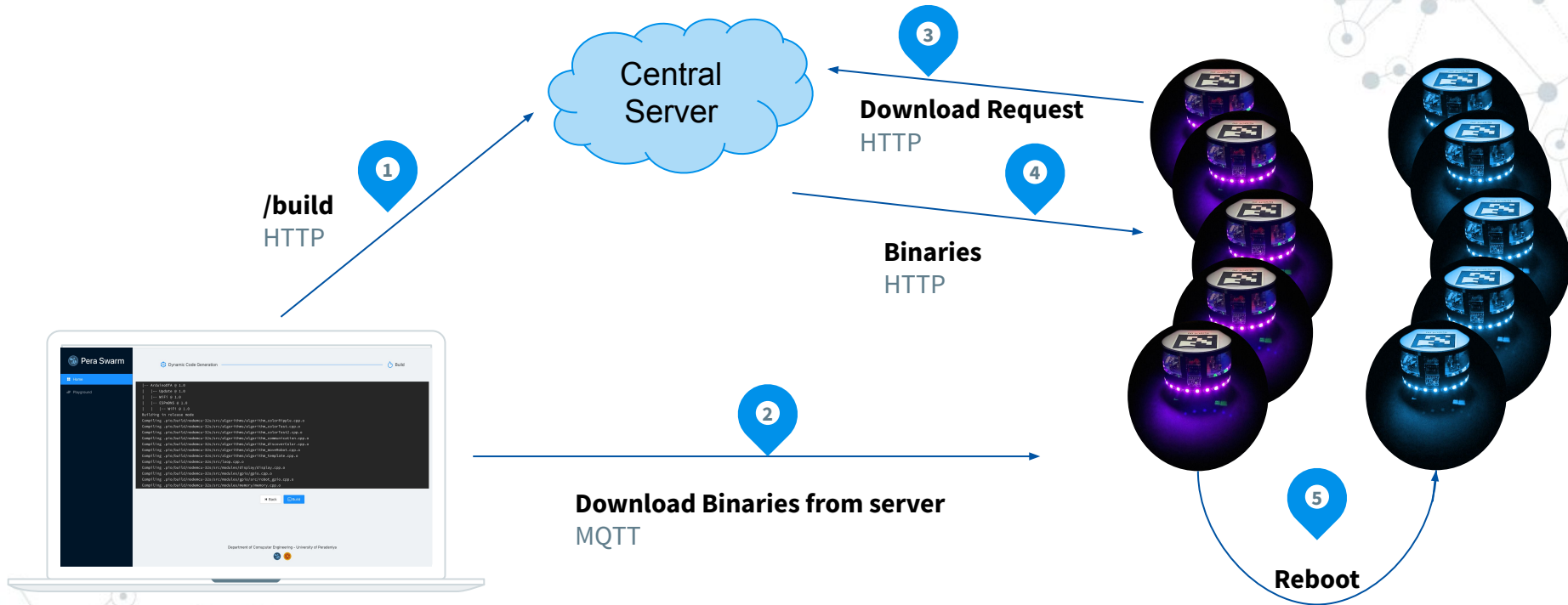
Remote Cross
Compiler
Development

1

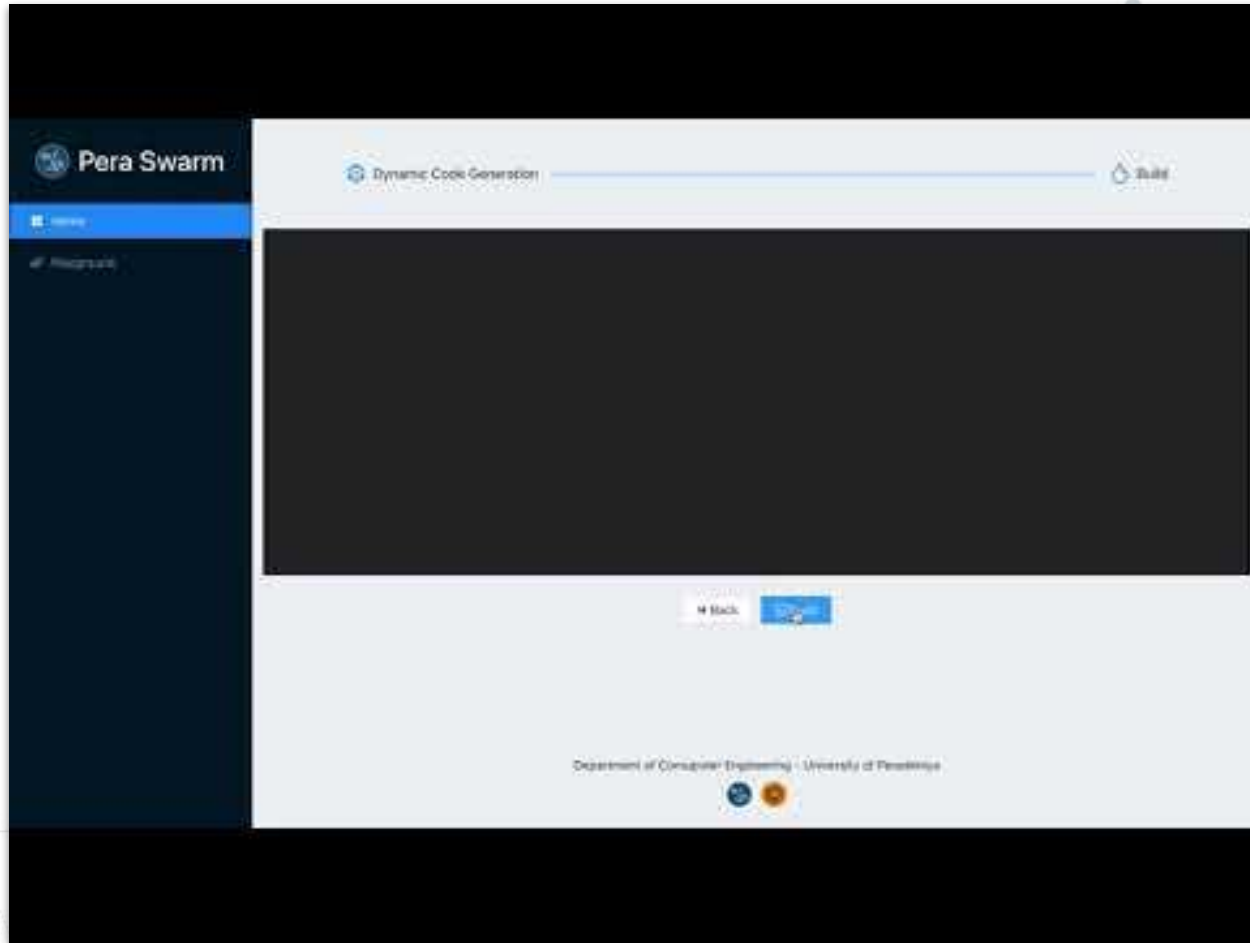
2

Code Generation & Uploading to
Robots

OTA (Over The Air) Upload



Code Generation & Uploading to Robots (I)



Roadmap

Remote Cross
Compiler
Development

1

3

Algorithms Development &
Testing

2

Code Generation &
Uploading to
Robots(I)

Algorithms Development & Testing

The screenshot displays the Pera Swarm IDE interface. On the left is a dark sidebar with a 'Pera Swarm' logo and navigation options: 'Home' (selected) and 'MQTT'. The main workspace is titled 'Algorithm Generation' and contains a 'Next' button. A vertical menu on the left lists categories: Logic, Algorithm, Control Flow, Inputs, MQTT, Motors, Outputs, Variables, String, Math, Functions, and ESP. The central workspace shows a block-based algorithm with the following structure:

```
color_change
ROBOT_STATE = 1
algorithm_loop
  if ROBOT_STATE == 0
  do
    algorithm_execute
    Delay 50
  else if ROBOT_STATE == 1
  do
    algorithm_setup
    set ROBOT_STATE to 10
  else
    Delay 100
```

Surrounding the main loop are several sub-algorithms:

- algorithm_setup:** Serial Print algorithm: setup
- algorithm_start:** Serial Print algorithm: start, set ROBOT_STATE to 0
- algorithm_reset:** Serial Print algorithm: reset, set ROBOT_STATE to 1
- algorithm_stop:** Serial Print algorithm: stop, set ROBOT_STATE to 10
- algorithm_execute:** Serial Print algorithm: execute, NeoPixel colorwave Red, Delay 1000, NeoPixel colorwave Green, Delay 1000, NeoPixel colorwave White, Delay 1000
- algorithm_interrupt:** robot_interrupt_t interrupt, char msg

A decorative network diagram in the top-left corner, consisting of various sized nodes (some solid grey, some hollow white) connected by thin grey lines, forming a complex web-like structure.

8

Demonstration

Home

The screenshot displays the Pera Swarm algorithm editor interface. On the left, a sidebar lists categories: Logic, Algorithm, Control Flow, Inputs, WDT, Motors, Outputs, Variables, String, Math, Functions, and SOP. The 'Functions' category is selected, showing blocks for 'Algorithm_start', 'Algorithm_run', 'Algorithm_stop', 'Algorithm_save', 'Algorithm_load', and 'Algorithm_delete'. The main workspace contains several blocks: 'Algorithm_start' (purple), 'Send First Approximation' (yellow), 'set ROBOT_STATE to 0' (purple), 'set ROBOT_STATE to 1' (purple), 'set ROBOT_STATE to 2' (purple), 'set ROBOT_STATE to 3' (purple), 'set ROBOT_STATE to 4' (purple), 'set ROBOT_STATE to 5' (purple), 'set ROBOT_STATE to 6' (purple), 'set ROBOT_STATE to 7' (purple), 'set ROBOT_STATE to 8' (purple), 'set ROBOT_STATE to 9' (purple), 'set ROBOT_STATE to 10' (purple), 'set ROBOT_STATE to 11' (purple), 'set ROBOT_STATE to 12' (purple), 'set ROBOT_STATE to 13' (purple), 'set ROBOT_STATE to 14' (purple), 'set ROBOT_STATE to 15' (purple), 'set ROBOT_STATE to 16' (purple), 'set ROBOT_STATE to 17' (purple), 'set ROBOT_STATE to 18' (purple), 'set ROBOT_STATE to 19' (purple), 'set ROBOT_STATE to 20' (purple), 'set ROBOT_STATE to 21' (purple), 'set ROBOT_STATE to 22' (purple), 'set ROBOT_STATE to 23' (purple), 'set ROBOT_STATE to 24' (purple), 'set ROBOT_STATE to 25' (purple), 'set ROBOT_STATE to 26' (purple), 'set ROBOT_STATE to 27' (purple), 'set ROBOT_STATE to 28' (purple), 'set ROBOT_STATE to 29' (purple), 'set ROBOT_STATE to 30' (purple), 'set ROBOT_STATE to 31' (purple), 'set ROBOT_STATE to 32' (purple), 'set ROBOT_STATE to 33' (purple), 'set ROBOT_STATE to 34' (purple), 'set ROBOT_STATE to 35' (purple), 'set ROBOT_STATE to 36' (purple), 'set ROBOT_STATE to 37' (purple), 'set ROBOT_STATE to 38' (purple), 'set ROBOT_STATE to 39' (purple), 'set ROBOT_STATE to 40' (purple), 'set ROBOT_STATE to 41' (purple), 'set ROBOT_STATE to 42' (purple), 'set ROBOT_STATE to 43' (purple), 'set ROBOT_STATE to 44' (purple), 'set ROBOT_STATE to 45' (purple), 'set ROBOT_STATE to 46' (purple), 'set ROBOT_STATE to 47' (purple), 'set ROBOT_STATE to 48' (purple), 'set ROBOT_STATE to 49' (purple), 'set ROBOT_STATE to 50' (purple), 'set ROBOT_STATE to 51' (purple), 'set ROBOT_STATE to 52' (purple), 'set ROBOT_STATE to 53' (purple), 'set ROBOT_STATE to 54' (purple), 'set ROBOT_STATE to 55' (purple), 'set ROBOT_STATE to 56' (purple), 'set ROBOT_STATE to 57' (purple), 'set ROBOT_STATE to 58' (purple), 'set ROBOT_STATE to 59' (purple), 'set ROBOT_STATE to 60' (purple), 'set ROBOT_STATE to 61' (purple), 'set ROBOT_STATE to 62' (purple), 'set ROBOT_STATE to 63' (purple), 'set ROBOT_STATE to 64' (purple), 'set ROBOT_STATE to 65' (purple), 'set ROBOT_STATE to 66' (purple), 'set ROBOT_STATE to 67' (purple), 'set ROBOT_STATE to 68' (purple), 'set ROBOT_STATE to 69' (purple), 'set ROBOT_STATE to 70' (purple), 'set ROBOT_STATE to 71' (purple), 'set ROBOT_STATE to 72' (purple), 'set ROBOT_STATE to 73' (purple), 'set ROBOT_STATE to 74' (purple), 'set ROBOT_STATE to 75' (purple), 'set ROBOT_STATE to 76' (purple), 'set ROBOT_STATE to 77' (purple), 'set ROBOT_STATE to 78' (purple), 'set ROBOT_STATE to 79' (purple), 'set ROBOT_STATE to 80' (purple), 'set ROBOT_STATE to 81' (purple), 'set ROBOT_STATE to 82' (purple), 'set ROBOT_STATE to 83' (purple), 'set ROBOT_STATE to 84' (purple), 'set ROBOT_STATE to 85' (purple), 'set ROBOT_STATE to 86' (purple), 'set ROBOT_STATE to 87' (purple), 'set ROBOT_STATE to 88' (purple), 'set ROBOT_STATE to 89' (purple), 'set ROBOT_STATE to 90' (purple), 'set ROBOT_STATE to 91' (purple), 'set ROBOT_STATE to 92' (purple), 'set ROBOT_STATE to 93' (purple), 'set ROBOT_STATE to 94' (purple), 'set ROBOT_STATE to 95' (purple), 'set ROBOT_STATE to 96' (purple), 'set ROBOT_STATE to 97' (purple), 'set ROBOT_STATE to 98' (purple), 'set ROBOT_STATE to 99' (purple), 'set ROBOT_STATE to 100' (purple). The blocks are arranged in a sequence, with some blocks having a 'set ROBOT_STATE to' value. The blocks are connected by lines, indicating a flow of execution. The interface is clean and modern, with a dark blue sidebar and a light blue main workspace.



9

Challenges

Challenges and How overcame them

Challenges



Dependency issue when building the existing firmware of robots.



Google Blockly visual programming tools doesn't support c++ by default.



Need to upload specific firmware to each robot by OTA.



Thanks!
Any questions?

