|  |  |
| --- | --- |
| **Logo_FPT_University_doc** | **MINISTRY OF EDUCATION AND TRAINING** |

|  |
| --- |
| **FPT UNIVERSITY** |
| Capstone Project Document |
| Designing and making  A Lynxmotion A-POD robot controller |
|  |
| |  |  | | --- | --- | | **Hexapod Team** | | | **Group Members** | * Phan Anh Dũng Cường * Nguyễn Minh Quân | | **Supervisor** | M.Si. Trần Khánh Ninh | | **Ext Supervisor** |  | | **Capstone Project code** | APOD | |
|  |

|  |
| --- |
| - Ho Chi Minh City, 09/2013 - |

# Table of Contents

[Table of Contents 2](#_Toc373452465)

[List of Figures 4](#_Toc373452466)

[List of Tables 6](#_Toc373452467)

[A. INTRODUCTION 9](#_Toc373452468)

[I. PROJECT INFORMATION 9](#_Toc373452469)

[II. TEAM MEMBER 9](#_Toc373452470)

[III. APOD INTRODUCTION 9](#_Toc373452471)

[1. HEXAPOD 9](#_Toc373452472)

[2. LYNXMOTION APOD 11](#_Toc373452473)

[IV. EXISTING SOLUTIONS 12](#_Toc373452474)

[V. PROJECT INTRODUCTION 13](#_Toc373452475)

[VI. APPROACHES 14](#_Toc373452476)

[VII. OTHER 14](#_Toc373452477)

[1. WHY WE NEED APOD ROBOT? 14](#_Toc373452478)

[2. PRODUCT 14](#_Toc373452479)

[B. SOFTWARE PROJECT MANAGEMENT PLAN (SPMP) 15](#_Toc373452480)

[I. PROBLEM DEFINITION 15](#_Toc373452481)

[1. NAME OF THIS CAPSTONE PROJECT 15](#_Toc373452482)

[2. PROBLEM ABSTRACT 15](#_Toc373452483)

[3. PROBLEM OVERVIEW 15](#_Toc373452484)

[II. PROJECT ORGANIZATION 17](#_Toc373452485)

[1. SOFTWARE PROCESS MODEL 17](#_Toc373452486)

[2. ROLES AND RESPONSIBILITIES 18](#_Toc373452487)

[3. TOOL AND TECHNIQUES 18](#_Toc373452488)

[III. PROJECT MANAGEMENT PLAN 19](#_Toc373452489)

[1. Task 19](#_Toc373452490)

[2. Task sheet 21](#_Toc373452491)

[3. All meeting minutes 22](#_Toc373452492)

[IV. CODING CONVENTION 22](#_Toc373452493)

[C. SOFTWARE REQUIREMENTS SPECIFICATION (SRS) 23](#_Toc373452494)

[I. USER REQUIREMENT SPECIFICATIONS 23](#_Toc373452495)

[II. SYSTEM REQUIREMENT SPECIFICATIONS 23](#_Toc373452496)

[1. Hardware requirement 23](#_Toc373452497)

[2. Software interface 23](#_Toc373452498)

[3. External interface 23](#_Toc373452499)

[4. Functional requirement 23](#_Toc373452500)

[5. Non-functional requirement 24](#_Toc373452501)

[III. USE-CASE DIAGRAM AND USE-CASE SPECIFICATIONS 24](#_Toc373452502)

[1. Use-case Diagram 24](#_Toc373452503)

[2. Use-case Specifications 25](#_Toc373452504)

[IV. SOFTWARE SYSTEM ATTRIBUTES 51](#_Toc373452505)

[1. Reliability 51](#_Toc373452506)

[2. Availability 52](#_Toc373452507)

[3. Security 52](#_Toc373452508)

[4. Maintainability 52](#_Toc373452509)

[5. Portability 52](#_Toc373452510)

[6. Performance 52](#_Toc373452511)

[V. ENTITY RELATIONSHIP DIAGRAM 53](#_Toc373452512)

[D. SOFTWARE DESIGN DESCRIPTION (SDD) 53](#_Toc373452513)

[I. DESIGN OVERVIEW 53](#_Toc373452514)

[1. Introduction 53](#_Toc373452515)

[2. Architecture 54](#_Toc373452516)

[3. User interface 54](#_Toc373452517)

[II. COMPONENTS 55](#_Toc373452518)

[1. Central processing: 55](#_Toc373452519)

[2. Servos control & Sensor reader 57](#_Toc373452520)

[III. IMPLEMENTATION APPROACHES 59](#_Toc373452521)

[1. Algorithms and Mechanism 59](#_Toc373452522)

[2. SSC32 Board 73](#_Toc373452523)

[3. STM32F4 Discovery Board 75](#_Toc373452524)

[IV. ACTIVITY AND CLASS STRUCTURE 75](#_Toc373452525)

[1. Activity 75](#_Toc373452526)

[2. Class Structure 79](#_Toc373452527)

[V. SEQUENCE DIAGRAM 80](#_Toc373452528)

[1. Embedded Controller 80](#_Toc373452529)

[2. Use-cases Sequences 84](#_Toc373452530)

[VI. USER INTERFACE DESIGN 97](#_Toc373452531)

[1. Live Control 97](#_Toc373452532)

[2. Sequence Player 98](#_Toc373452533)

[3. Configuration 99](#_Toc373452534)

[4. Object Extractor 100](#_Toc373452535)

[E. SOFTWARE TEST DOCUMENTATION (STD) 101](#_Toc373452536)

[F. SOFTWARE USER’S MANUAL (SUM) 101](#_Toc373452537)

[G. APPENDIX 101](#_Toc373452538)

[I. Glossary 101](#_Toc373452539)

[II. References 101](#_Toc373452540)

# List of Figures

[Figure A‑1 CH3-R Walking Robot 10](#_Toc373452630)

[Figure A‑2 AH2 Walking Robot 10](#_Toc373452631)

[Figure A‑3 Lynxmotion APOD 11](#_Toc373452632)

[Figure A‑4 Flowbotics Studio example 1 12](#_Toc373452633)

[Figure A‑5 Flowbotics Studio example 2 12](#_Toc373452634)

[Figure A‑6 Lynxmotion Visual Sequences 13](#_Toc373452635)

[Figure B‑1 System Boundary Overview 16](#_Toc373452636)

[Figure B‑2 Spriral model 17](#_Toc373452637)

[Figure B‑3 Project plan 22](#_Toc373452638)

[Figure B‑4 Project plan 22](#_Toc373452639)

[Figure C‑1: Use cases diagram 24](#_Toc373452640)

[Figure C‑2 Add State Use-case diagram 25](#_Toc373452641)

[Figure C‑3 Remove States Use-case diagram 26](#_Toc373452642)

[Figure C‑4 Import States Use-case diagram 27](#_Toc373452643)

[Figure C‑5 Export States Use-case diagram 29](#_Toc373452644)

[Figure C‑6 Normal input Use-case diagram 30](#_Toc373452645)

[Figure C‑7 Gamepad Input Use-case diagram 31](#_Toc373452646)

[Figure C‑8 Move Forward Use-case diagram 33](#_Toc373452647)

[Figure C‑9 Move Backward Use-case diagram 34](#_Toc373452648)

[Figure C‑10 Turn Left Use-case diagram 36](#_Toc373452649)

[Figure C‑11 Turn Right Use-case diagram 37](#_Toc373452650)

[Figure C‑12 Body Lift Use-case diagram 39](#_Toc373452651)

[Figure C‑13 Body Drop Use-case diagram 40](#_Toc373452652)

[Figure C‑14 Squeeze Left Use-case diagram 41](#_Toc373452653)

[Figure C‑15 Squeeze Right Use-case diagram 43](#_Toc373452654)

[Figure C‑16 Manual Grip Use-case diagram 44](#_Toc373452655)

[Figure C‑17 Auto Grip Use-case diagram 46](#_Toc373452656)

[Figure C‑18 Choose Target Use-case diagram 47](#_Toc373452657)

[Figure C‑19 Change Target Use-case diagram 49](#_Toc373452658)

[Figure C‑20 Lock Target Use-case diagram 50](#_Toc373452659)

[Figure C‑21 53](#_Toc373452660)

[Figure D‑1 APOD Controller System Architecture 54](#_Toc373452661)

[Figure D‑2 Live Control User interface design 54](#_Toc373452662)

[Figure D‑3 Sequence Player user interface design 55](#_Toc373452663)

[Figure D‑4 Bluetooth Communication module HC05 56](#_Toc373452664)

[Figure D‑5 STM32F4 Discovery Board 57](#_Toc373452665)

[Figure D‑6 HC 645MG Servo 58](#_Toc373452666)

[Figure D‑7 SSC32 Board 58](#_Toc373452667)

[Figure D‑8 HCSR04 Distance Sensor 59](#_Toc373452668)

[Figure D‑9 RC Servo block 60](#_Toc373452669)

[Figure D‑10 Servo Signal illustration 60](#_Toc373452670)

[Figure D‑11 HC 645MG Signal Details 61](#_Toc373452671)

[Figure D‑12 HC-SR04 Specifications 62](#_Toc373452672)

[Figure D‑13 HC-SR04 Best practice angle 62](#_Toc373452673)

[Figure D‑14 HC-SR04 Operating sequence 63](#_Toc373452674)

[Figure D‑15 Tripod Gait illustration 64](#_Toc373452675)

[Figure D‑16 Wave Gait illustration 65](#_Toc373452676)

[Figure D‑17 Wave Gait step order 65](#_Toc373452677)

[Figure D‑18 Ripple Gait illustration 66](#_Toc373452678)

[Figure D‑19 APOD leg position block 67](#_Toc373452679)

[Figure D‑20 APOD leg swing diagram 68](#_Toc373452680)

[Figure D‑21 APOD Leg full step diagram 68](#_Toc373452681)

[Figure D‑22 Tripod A at Front position 69](#_Toc373452682)

[Figure D‑23 Tripod A at Neutral position 70](#_Toc373452683)

[Figure D‑24 Tripod A at Rear position 70](#_Toc373452684)

[Figure D‑25 Tripod A at Left position 72](#_Toc373452685)

[Figure D‑26 Tripod A at Right position 72](#_Toc373452686)

[Figure D‑27 SSC32 Schematic 74](#_Toc373452687)

[Figure D‑28 76](#_Toc373452688)

[Figure D‑29 77](#_Toc373452689)

[Figure D‑30 78](#_Toc373452690)

[Figure D‑31 79](#_Toc373452691)

[Figure D‑32 80](#_Toc373452692)

[Figure D‑33 81](#_Toc373452693)

[Figure D‑34 82](#_Toc373452694)

[Figure D‑35 83](#_Toc373452695)

[Figure D‑36 84](#_Toc373452696)

[Figure D‑37 85](#_Toc373452697)

[Figure D‑38 85](#_Toc373452698)

[Figure D‑39 86](#_Toc373452699)

[Figure D‑40 87](#_Toc373452700)

[Figure D‑41 88](#_Toc373452701)

[Figure D‑42 89](#_Toc373452702)

[Figure D‑43 89](#_Toc373452703)

[Figure D‑44 90](#_Toc373452704)

[Figure D‑45 90](#_Toc373452705)

[Figure D‑46 91](#_Toc373452706)

[Figure D‑47 91](#_Toc373452707)

[Figure D‑48 92](#_Toc373452708)

[Figure D‑49 92](#_Toc373452709)

[Figure D‑50 93](#_Toc373452710)

[Figure D‑51 94](#_Toc373452711)

[Figure D‑52 95](#_Toc373452712)

[Figure D‑53 96](#_Toc373452713)

[Figure D‑54 Live Control Interface Design 97](#_Toc373452714)

[Figure D‑55 Sequence Player Interface Design 98](#_Toc373452715)

[Figure D‑56 Configuration Interface Design 99](#_Toc373452716)

[Figure D‑57 Object Extractor Interface Design 100](#_Toc373452717)

# List of Tables

[Table A‑1 Team Members 9](#_Toc373452718)

[Table B‑1 Roles and Responsibilities 18](#_Toc373452719)

[Table B‑2 Initiating and Planning 19](#_Toc373452720)

[Table B‑3 Software Requirement Analysis 19](#_Toc373452721)

[Table B‑4 Creating Software Design Description 20](#_Toc373452722)

[Table B‑5 Coding 20](#_Toc373452723)

[Table B‑6 Testing 20](#_Toc373452724)

[Table B‑7 Deployment 21](#_Toc373452725)

[Table C‑1 26](#_Toc373452726)

[Table C‑2 27](#_Toc373452727)

[Table C‑3 29](#_Toc373452728)

[Table C‑4 30](#_Toc373452729)

[Table C‑5 31](#_Toc373452730)

[Table C‑6 33](#_Toc373452731)

[Table C‑7 34](#_Toc373452732)

[Table C‑8 35](#_Toc373452733)

[Table C‑9 37](#_Toc373452734)

[Table C‑10 38](#_Toc373452735)

[Table C‑11 40](#_Toc373452736)

[Table C‑12 41](#_Toc373452737)

[Table C‑13 43](#_Toc373452738)

[Table C‑14 44](#_Toc373452739)

[Table C‑15 46](#_Toc373452740)

[Table C‑16 47](#_Toc373452741)

[Table C‑17 48](#_Toc373452742)

[Table C‑18 50](#_Toc373452743)

[Table C‑19 51](#_Toc373452744)

[Table D‑1 67](#_Toc373452745)

[Table D‑2 APOD Leg full forward step 69](#_Toc373452746)

[Table D‑3 Forward Sequence 71](#_Toc373452747)

[Table D‑4 Backward Sequence 71](#_Toc373452748)

[Table D‑5 Turn Left Sequence 73](#_Toc373452749)

[Table D‑6 Turn Right Sequence 73](#_Toc373452750)

[Table D‑7 Live Control Interface Design Description 97](#_Toc373452751)

[Table D‑8 Sequence Player Interface Design Description 98](#_Toc373452752)

[Table D‑9 Configuration Interface Design Description 99](#_Toc373452753)

[Table D‑10 Object Extractor Interface Design Description 100](#_Toc373452754)

# INTRODUCTION

## PROJECT INFORMATION

* Project name: **Lynxmotion A-Pod  robot controller**
* Project code: **APOD**
* Product type: **Embedded robot controller**
* Timeline: **from September 2013 to December 2013**

## TEAM MEMBER

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Name** | **Role** | **Contact** |
| **1** | Phan Anh Dũng Cường | Leader | * Phone: 0972708318 * Email: cuongpadse60774@fpt.edu.vn |
| **2** | Nguyễn Minh Quân | Member | * Phone: 0915699635 * Email: quannm60344@fpt.edu.vn |
|  |  |  |  |
|  |  |  |  |

Table A‑1 Team Members

## APOD INTRODUCTION

### HEXAPOD

The trademarked name “hexapod” (by Geodetic Technology) was originally for Stewart platforms used in machine tools. However, the term is now used for 6-jack platforms outside of the machine tool area, since it simply means "six legs".

In the scope of this document, “hexapod” will represent a structure with six legs which can be used for variety of Robotics application.

Variety of robotics hexapod:



Figure A‑1 CH3-R Walking Robot

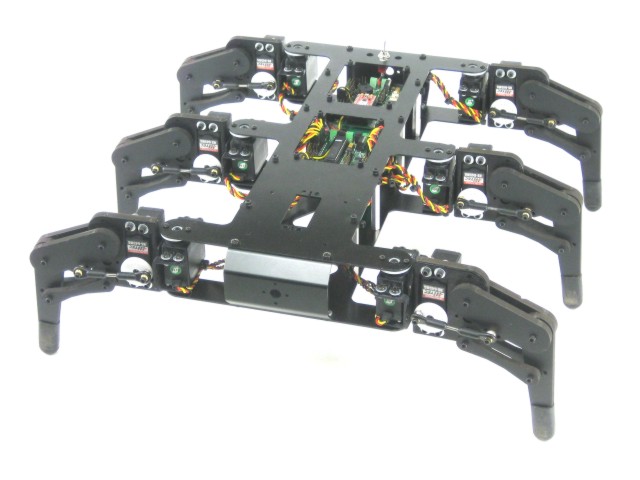


Figure A‑2 AH2 Walking Robot

### LYNXMOTION APOD



Figure A‑3 Lynxmotion APOD

* Insect inspired body.
* Assembly with 25 servos (motors) and PVC materials to form 6 legs, mandible grippers (with force sensor feature) and a tail.
* The three DOF (degree of freedom) leg design help the APOD moving in a variety of directions.
* Servos are directly controlled by SSC-32 board which receive command via RS232 interface from Bot board II / PC or any other devices communicate using RS232.

## EXISTING SOLUTIONS

* Flowbotics Studio: $39.9

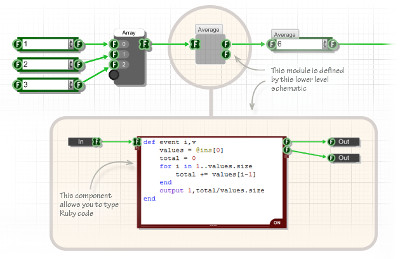


Figure A‑4 Flowbotics Studio example 1

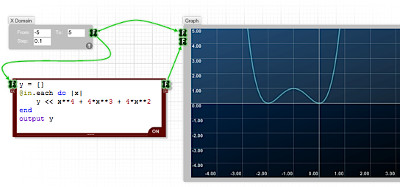


Figure A‑5 Flowbotics Studio example 2

* Lynxmotion Visual Sequences: $39.94

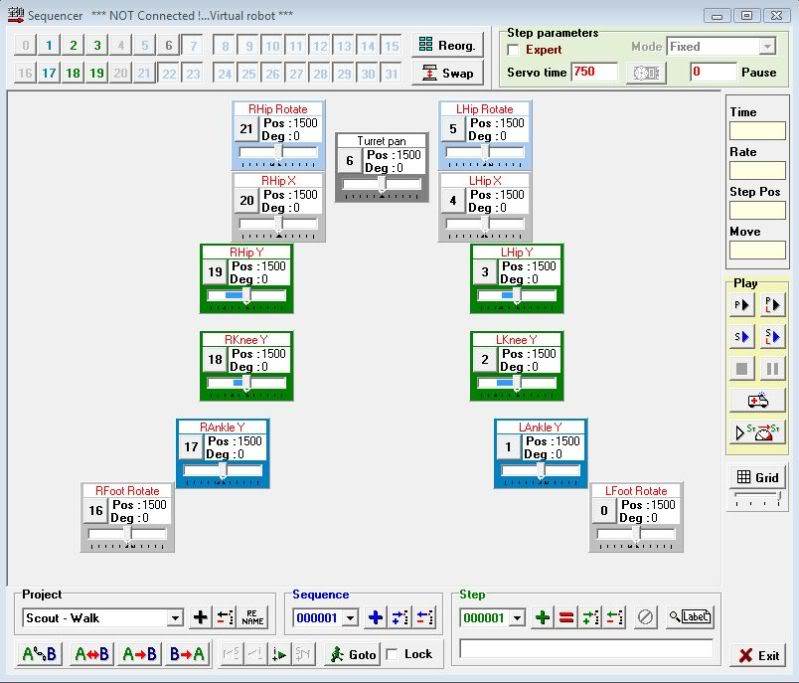


Figure A‑6 Lynxmotion Visual Sequences

## PROJECT INTRODUCTION

This project focus on programming on microprocessor unit (MCU) to developing, control the A-Pod robot. The BASIC is let him move forward, backward, right, left. Thus, we concern on control over Bluetooth. Through Camera put on robot control it/ or automatic function, connect with Sensor to discover obstacles.

Abstracts purpose: Design the controller board motor servo using microcontroller to control Lynxmotion A-Pod  robot  with available mechanical. Implement the hardware and software on PC to control Lynxmotion. The system requires Microcontroller ARM LPC,AT91SAM,PIC,AVR. Finding critical points and find best way algorithms.

## APPROACHES

* Building SSC32 firmware with ATMega328p microcontroller to control 25 servos.
* Using Bot Board II as universal sensor reader to serve automation purposes (additional sensors will be needed).
* Additional STM32F4 Discovery using ARM will be used as a central processing module which communicate with other device via Bluetooth.
* IP Camera to provide an insight view for user.
* A friendly HMI program will be provided for user to control the APOD with ease.

## OTHER

### WHY WE NEED APOD ROBOT?

The main purpose of A-Pod Robot is to be used at home with variable simple tasks. The Project ‘s scope is to control A-Pod from a distance via a HCI program on PC (or mobile devices). A-Pod can perform task without the present of human and report back working information when needed.

### PRODUCT

* HMI Program on PC (Possible extend: Android/IOS).
* Servo Control Module Software (SSC32 firmware).
* Sensor Service Module Software (Bot Board II firmware)
* Central Processing Module Software (STM32F4 Discovery).

# SOFTWARE PROJECT MANAGEMENT PLAN (SPMP)

## PROBLEM DEFINITION

### NAME OF THIS CAPSTONE PROJECT

* + Project name: **Lynxmotion A-Pod  robot controller**
  + Project code: **APOD**
  + Product type: **Embedded robot controller**

### PROBLEM ABSTRACT

As mentioned above, this project will focus on implement a new control system for APOD using Bluetooth/wireless. The main purpose of new controller system is to make the APOD more flexible. In order to achieve that purpose, the final product should have those characteristics:

* + Friendly HMI interface.
  + Easy to use, easy to learn.
  + Different user experience: comfort and high interaction.
  + Adaptable to new mechanical constraint, prepared for complete automation.

### PROBLEM OVERVIEW

#### The Current System

Current APOD control system can be control using either USB-to-serial cable or an PS2 interface. The SSC32 will be responsible for direct controlling of motor servos.

* Advantages:
  + Direct command.
  + No delay due to no-subsystem needed (USB-to-serial control).
  + Simple architecture
* Limitation:
  + Very limited range of control due to “wire” problem
  + A considerable amount of wire/cable can be obstacles for APOD movement.
  + Domain or mechanical knowledge is required to perform.
  + On-sight performance: APOD must stay insight of user viewpoint.

#### The Propose System

The system consists of 4 module:

* + Servos Control Module (SSC-32): Directly control servo ‘s movement.
  + Sensor Service Module (Bot Board II): receive all kind of sensor signal and value.
  + Central Processing Module (STM32F4 Discovery) plus Bluetooth module
  + HMI software on PC

Main features will be provided:

* + Basic movement: forward, backward, turn left, turn right.
  + High interactive user interface.

#### Boundaries of The System

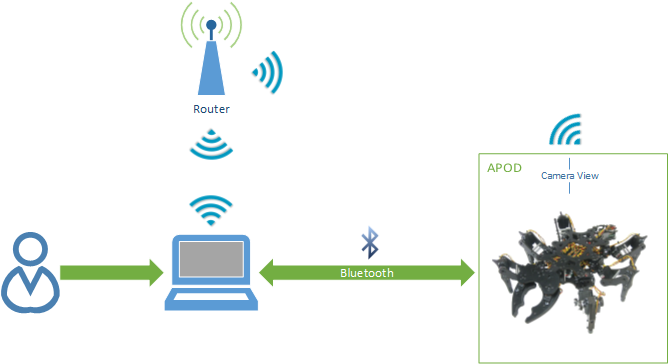


Figure B‑1 System Boundary Overview

* + The APOD can perform only within Bluetooth or Wireless device ‘s range
  + Any interfere while receiving command (transmitting via Bluetooth) will cause the APOD to misbehave or unpredictable movement.

#### Developing Environment

##### Hardware environment:

* Lyxnmotion APOD
* IP camera, Bluetooth device, Distance sensor.
* 4 laptops with appropriate configuration for embedded development.

##### Software environment:

* Developing environment ARV studio 4 for main board programming.
* Developing environment Visual studio 2012 with C# language for user interface programming.

## PROJECT ORGANIZATION

### SOFTWARE PROCESS MODEL

The process model used for developing this project is Spiral Model.

The spiral model is a software development process combining elements of both design and prototyping-in-stages, in an effort to combine advantages of top-down and bottom-up concepts. Also known as the spiral lifecycle model (or spiral development), it is a systems development method (SDM) used in information technology (IT). This model of development combines the features of the prototyping and the waterfall model. The spiral model is intended for large, expensive and complicated projects.



Figure B‑2 Spriral model

### ROLES AND RESPONSIBILITIES

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Full name | Team Role | Responsibilities |
| 1 | Trần Khánh Ninh | Supervisor | Define business  Support in technical issues |
| 1 | Phan Anh Dũng Cường | Team Leader | Prioritize work  Answer question and deliver direction  Facilitate productivity – maximize team performance  Complete all individual work |
| 3. | Nguyễn Minh Quân | Team Member | Commit individual product on time  Support each other to complete team work |
|  |  |  |  |

Table B‑1 Roles and Responsibilities

### TOOL AND TECHNIQUES

#### For Development

##### Hardware environment:

* Lynxmotion APOD
* IP camera, Bluetooth device, Distance sensor.
* 4 laptops with appropriate configuration for embedded development.

##### Software environment:

* Developing environment ARV studio 4 for main board programming.
* Developing environment Visual studio 2010 with C# language for user interface programming.
* Developing environment Keilc studio for program ARM.

#### For Management

* Microsoft Project: Task tracking
* Tortoise SVN: Source version control
* Google Code: Connect, synchronize source code and documents
* Beyond Compare: Manage changes in source files

#### For Communication

* Gmail report
* Skype chat system
* CMS system

## PROJECT MANAGEMENT PLAN

### Task

#### Initiating and Planning:

|  |  |
| --- | --- |
| Description | Registering project, kick-off meeting and planning |
| Output | Registered project, team spirit, overview plan |
| Deliverables | Draft project plan |
| Effort (man-month[[1]](#footnote-1)) | 0.8 mm |
| Dependencies and Constrains | N/A |
| Risks | Some members may be absent |

Table B‑2 Initiating and Planning

#### Software Requirement Analysis

|  |  |
| --- | --- |
| Description | Analyzing software requirements based on available hexapod’s hardware to create software requirements specification document |
| Output | Software Requirement Specification document |
| Deliverables | SRS document file |
| Effort (man-month) | 1.5 mm |
| Dependencies and Constrains | N/A |
| Risks | - Lack of knowledge about hexapod’s hardware |

Table B‑3 Software Requirement Analysis

#### Creating Software Design Description

|  |  |
| --- | --- |
| Description | Designing the controller for hexapod robot based on actual requirements |
| Output | Architecture design, circuits diagram, board diagram, algorithms and design specification |
| Deliverables | SDD document |
| Effort (man-month) | 3.0 mm |
| Dependencies and Constrains | Completion of SRS |
| Risks | - Choosing inappropriate algorithms and design patterns  - The hardware is hard to maintain  - Causing high coding efforts |

Table B‑4 Creating Software Design Description

#### Coding

|  |  |
| --- | --- |
| Description | Designing the controller for hexapod robot based on actual requirements |
| Output | Architecture design, circuits diagram, board diagram, algorithms and design specification |
| Deliverables | SDD document |
| Effort (man-month) | 3.0 mm |
| Dependencies and Constrains | Completion of SRS |
| Risks | - Choosing inappropriate algorithms and design patterns  - The hardware is hard to maintain  - Causing high coding efforts |

Table B‑5 Coding

#### Testing

|  |  |
| --- | --- |
| Description | Creating test case and execute test |
| Output | Test plan, test case document, test report, all tested modules and tested system |
| Deliverables | Test plan, test report |
| Effort (man-month) | 2.2 mm |
| Dependencies and Constrains | Completion of SRS, SDD, coding |
| Risks | * Lack of professional testers in team * Unit test may not be performed thoroughly causing spending many efforts in system test phase.   - Hardware ‘s limitations and errors  - ES testing is different with IS testing |

Table B‑6 Testing

#### Deployment

|  |  |
| --- | --- |
| Description | Deploying system include : hexapod robot and hexapod controller simulator |
| Output | Software packages, user manual |
| Deliverables | Software packages, user manual |
| Effort (man-month) | 0.3 mm |
| Dependencies and Constrains | Completion of all other tasks |
| Risks | - Hardware’s errors |

Table B‑7 Deployment

### Task sheet

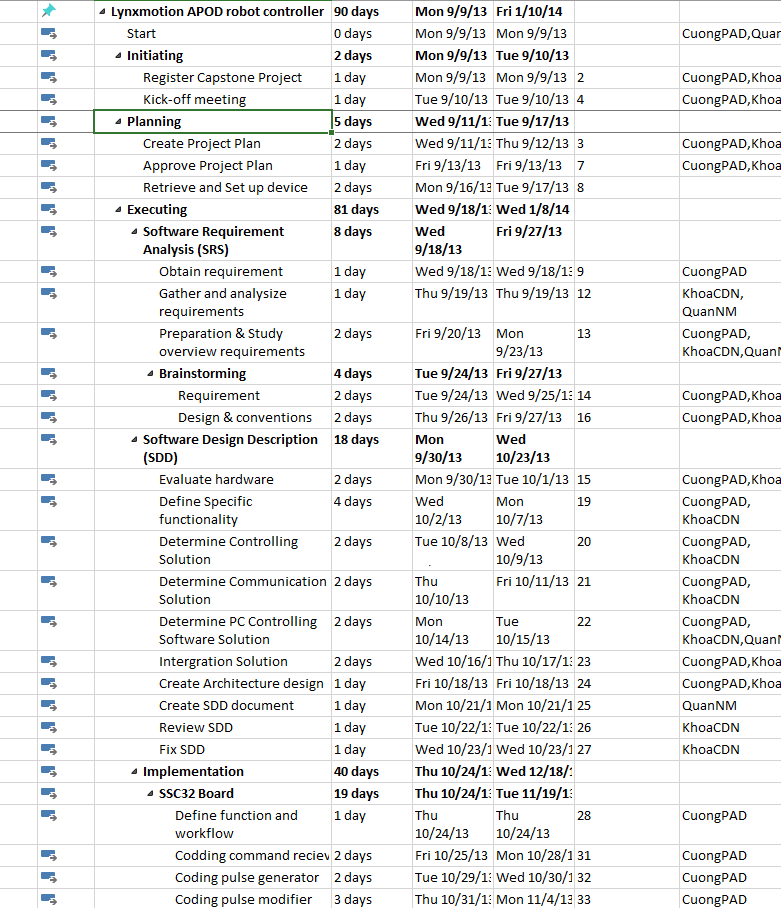


Figure B‑3 Project plan



Figure B‑4 Project plan

### All meeting minutes

<to be updated>

## CODING CONVENTION

The following rules follow:

* The standard rules for developing application using C/C++ (<http://users.ece.cmu.edu/~eno/coding/CCodingStandard.html>)

# SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

## USER REQUIREMENT SPECIFICATIONS

* A-Pod should be able to walk freely in any direction.
* Controlling via a Serial connection like Bluetooth, or directly through a PS2 controller.
* A-Pod should be able to grab small things like a Coke.
* A-Pod should be able to recording video.
* A-Pod should be able to detect objects stand (within 30cm) in the ways and response back.
* Controller should be simple and easy.
* After receive movement signal, A-Pod should move within 2 seconds

## SYSTEM REQUIREMENT SPECIFICATIONS

### Hardware requirement

* 25 HS-645MG servo
* 1 SSC-32 servo controller
* 1 Bot Board II with Basic Atom Pro 28 microcontroller
* 1 HC-SR04 supersonic sensor
* 1 STM32F4 ARM
* Laptops with Bluetooth Device
* 1 PS2 controller
* Few serial cables, adapters , jumpers…

### Software interface

* Menu-driven design with : button, dialog box, screen.
* Button : movement button on the right, setting like PS2 controller.
* Dialog box : on the left, display A-Pod ‘s currently speed, object stand in way and distance.
* Screen : center of menu controller, display view from camera.

### External interface

* PS2 Controller

### Functional requirement

* Direction control : user can direct the A-Pod to turn left, right, go forward or backward.
* Camera view control : user can see the view sending by the A-Pod camera.
* Grab control : user can direct the A-Pod to grab things.
* Remote control : enable remote controlling, help the user control the A-Pod through a programmed Bluetooth device.

### Non-functional requirement

* The A-Pod should response and move after receiving order within 2 seconds.
* The longest range for leg movement should be 30 centimeter, 35 degree.
* Detected range should be larger than 2 meter.
* View from camera should be refresh every 0.5 second.
* The A-Pod must be compatible with the following batteries and chargers:
  + NiCad & Ni-MH Universal Smart Charger (USC-02)
  + Volt Ni-MH 2800mAh Battery Pack
* All cable, jumper, adapter should be corrected connect

## USE-CASE DIAGRAM AND USE-CASE SPECIFICATIONS

### Use-case Diagram

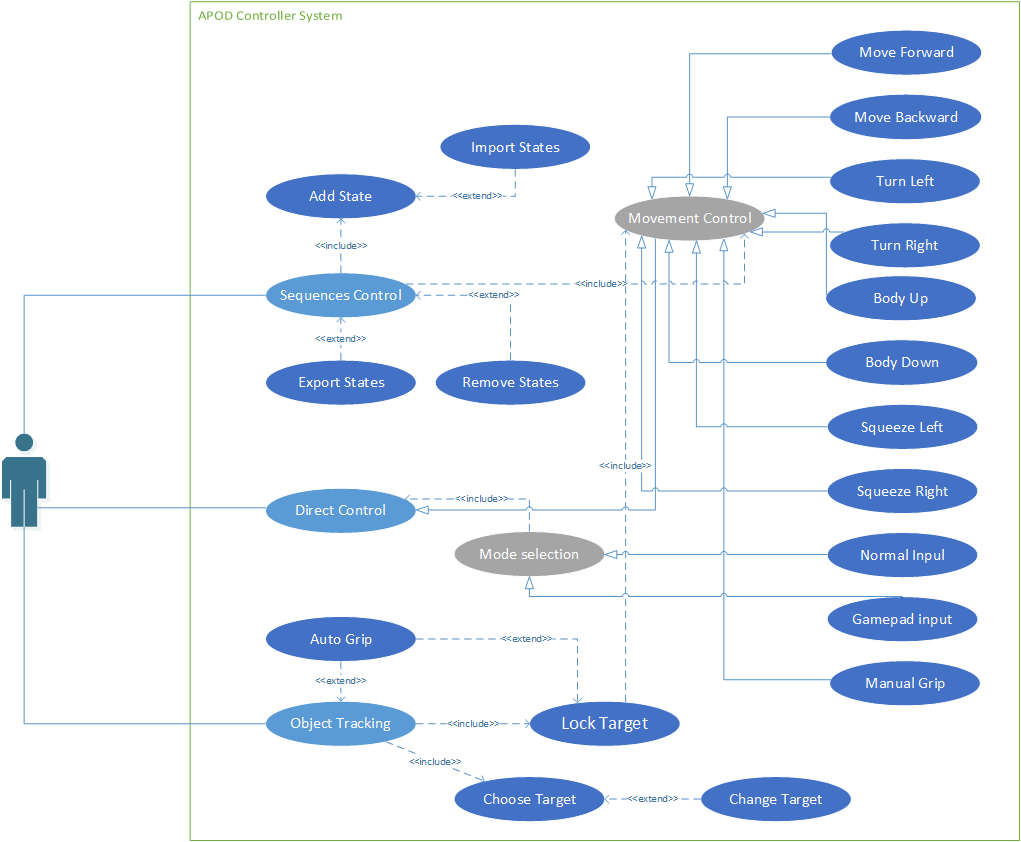
****

Figure C‑1: Use cases diagram

### Use-case Specifications

#### Sequences Control

##### Add State

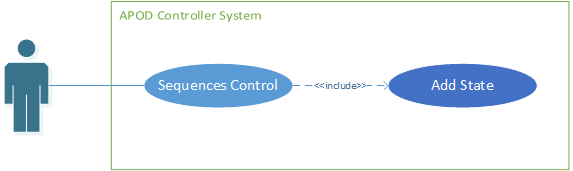


Figure C‑2 Add State Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ADD STATE USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC001 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Add State | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Add a state (move) to sequence  **Goal:**  New state is added to sequence  **Triggers**  User click [Add] button  **Preconditions:**   * Sequence Player feature is selected * Interval (if required) stay within valid range   **Post Conditions:**  NONE  **Main Success Scenario:**   * User select move to add to sequence * Enter interval value (if available) * Click [Add]   **Alternative Scenario:**  NONE  **Exceptions:**  NONE  **Relationships:**  NONE | | | | |

Table C‑1

##### Remove states

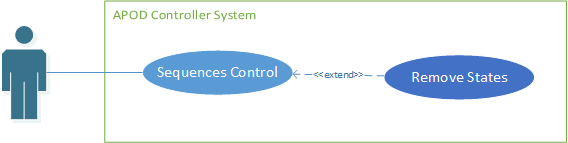


Figure C‑3 Remove States Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **REMOVE STATES USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC002 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Remove States | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Remove selected states from current sequence  **Goal:**  All selected state to be removed  **Triggers**  User click [Remove] Button  **Preconditions:**   * Sequence Player feature is selected   **Post Conditions:**  NONE  **Main Success Scenario:**   * User select all move to be removed from sequence * Click [Remove]   **Alternative Scenario:**  NONE  **Exceptions:**  NONE  **Relationships:**  NONE | | | | |

Table C‑2

##### Import States

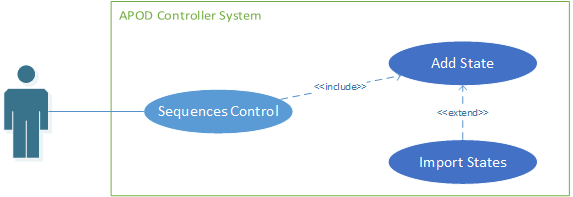


Figure C‑4 Import States Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **IMPORT STATES USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC003 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Import States | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Import existing sequence from file  **Goal:**  Current sequence is replace by new one from file  **Triggers**  User click [Import] Button  **Preconditions:**   * Sequence Player feature is selected   **Post Conditions:**  NONE  **Main Success Scenario:**   * Click [Import] button * Browse to location contain sequence file (\*.xml) * Select file & click [Ok] * Validate new sequence content * Replace current sequence if valid   **Alternative Scenario:**  NONE  **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑3

##### Export States

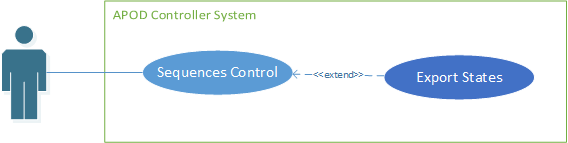


Figure C‑5 Export States Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EXPORT STATES USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC004 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Export States | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Export current sequence to file  **Goal:**  New Sequence file  **Triggers**  User click [Export] button  **Preconditions:**   * Sequence Player feature is selected   **Post Conditions:**  NONE  **Main Success Scenario:**   * Click [Export] button * Browse to location contain sequence file (\*.xml) * Click [Ok]   **Alternative Scenario:**  NONE  **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑4

#### Direct Control

##### Mode Selection

###### Normal Input

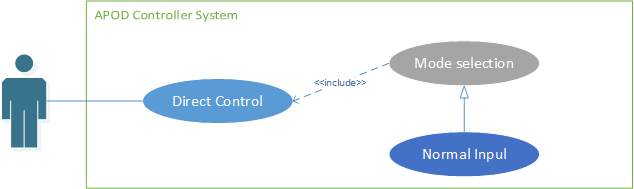


Figure C‑6 Normal input Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NORMAL INPUT USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC005 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Normal Input Selection | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Using normal input method (keyboard or Mouse)  **Goal:**  Normal input is selected  **Triggers**  User click [Normal Mode] button  **Preconditions:**   * Live control feature is selected   **Post Conditions:**  NONE  **Main Success Scenario:**   * User click [Normal Mode] button * Configuration is checked * Establish Bluetooth connection   **Alternative Scenario:**  NONE  **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑5

###### Gamepad Input

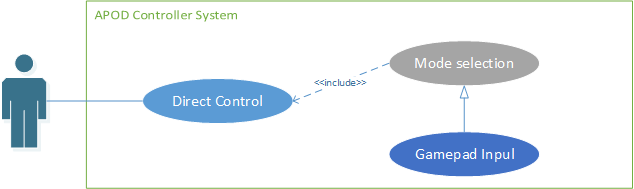


Figure C‑7 Gamepad Input Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **GAMEPAD INPUT** **USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC006 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Gamepad Input Selection | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Using Gamepad input method (Gamepad)  **Goal:**  Gamepad input is selected  **Triggers**  User click [Gamepad Mode] button  **Preconditions:**   * Live control feature is selected * Gamepad is connected   **Post Conditions:**  NONE  **Main Success Scenario:**   * User click [Gamepad Mode] button * Configuration is checked * Gamepad device connection is checked * Establish Bluetooth connection   **Alternative Scenario:**  NONE  **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑6

##### Movement Control

###### Move Forward

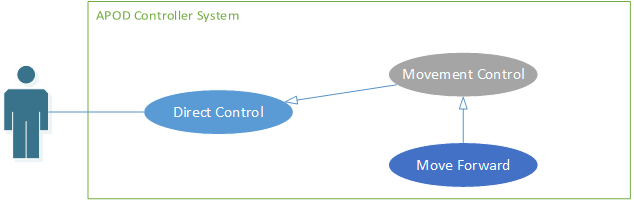


Figure C‑8 Move Forward Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MOVE FORWARD USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC007 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Move Forward | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Command APOD to move forward  **Goal:**  APOD walking in forward direction  **Triggers**  User click [Forward] button  **Preconditions:**   * Live control feature is selected * Bluetooth connection is established * Normal input mode or Gamepad input mode is selected * [L1] and [L2] is not pressed.   **Post Conditions:**   * APOD stop when button is released   **Main Success Scenario:**   * User click on [Forward] button ()   **Alternative Scenario:**   * Alt 1: press “W” key on keyboard (if Normal input is selected) * Alt 2: press () key on gamepad (if Gamepad input is selected)   **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑7

###### Move Backward

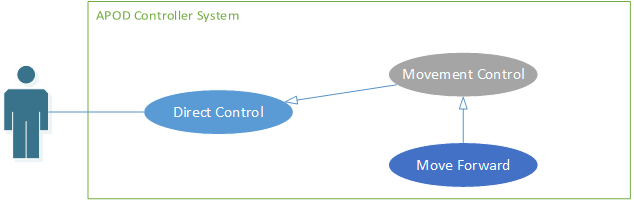


Figure C‑9 Move Backward Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MOVE BACKWARD USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC008 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Move Backward | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Command APOD to move Backward  **Goal:**  APOD walking in Backward direction  **Triggers**  User click [Backward] button  **Preconditions:**   * Live control feature is selected * Bluetooth connection is established * Normal input mode or Gamepad input mode is selected * [L1] and [L2] is not pressed.   **Post Conditions:**   * APOD stop when button is released   **Main Success Scenario:**   * User click on [Backward] button ()   **Alternative Scenario:**   * Alt 1: press “S” key on keyboard (if Normal input is selected) * Alt 2: press () key on gamepad (if Gamepad input is selected)   **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑8

###### Turn Left

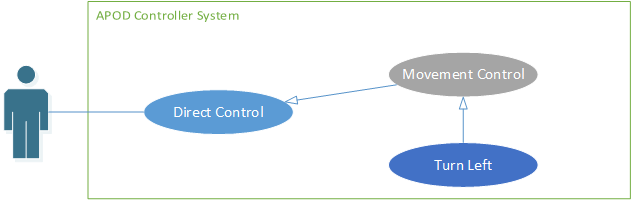


Figure C‑10 Turn Left Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TURN LEFT USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC009 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Turn Left | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Command APOD to turn to the left  **Goal:**  APOD turn in Left direction  **Triggers**  User click [Turn Left] button  **Preconditions:**   * Live control feature is selected * Bluetooth connection is established * Normal input mode or Gamepad input mode is selected * [L1] and [L2] is not pressed.   **Post Conditions:**   * APOD stop when button is released   **Main Success Scenario:**   * User click on [Turn Left] button ()   **Alternative Scenario:**   * Alt 1: press “A” key on keyboard (if Normal input is selected) * Alt 2: press () key on gamepad (if Gamepad input is selected)   **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑9

###### Turn Right

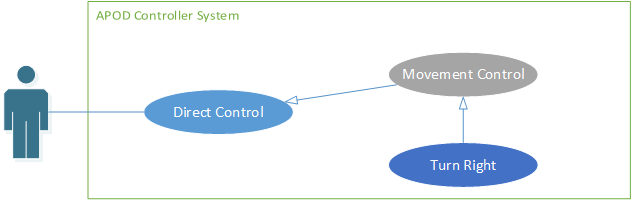


Figure C‑11 Turn Right Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TURN RIGHT USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC010 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Turn Right | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Command APOD to turn to the Right  **Goal:**  APOD turn in Right direction  **Triggers**  User click [Turn Right] button  **Preconditions:**   * Live control feature is selected * Bluetooth connection is established * Normal input mode or Gamepad input mode is selected * [L1] and [L2] is not pressed.   **Post Conditions:**   * APOD stop when button is released   **Main Success Scenario:**   * User click on [Turn Right] button ()   **Alternative Scenario:**   * Alt 1: press “D” key on keyboard (if Normal input is selected) * Alt 2: press () key on gamepad (if Gamepad input is selected)   **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑10

###### Body Lift

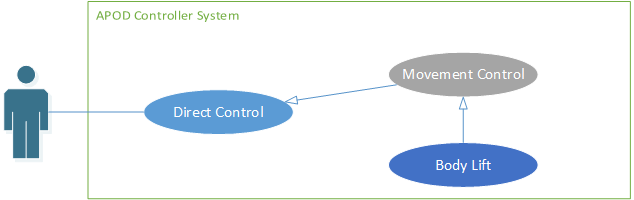


Figure C‑12 Body Lift Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **BODY LIFT USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC011 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Body Lift | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Command APOD to Lift up  **Goal:**  APOD Lift body up  **Triggers**  User click [R1] button  **Preconditions:**   * Live control feature is selected * Bluetooth connection is established * Normal input mode or Gamepad input mode is selected   **Post Conditions:**  NONE  **Main Success Scenario:**   * User click on [R1] button ()   **Alternative Scenario:**   * Alt 1: press “U” key on keyboard (if Normal input is selected) * Alt 2: press () key on gamepad (if Gamepad input is selected)   **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑11

###### Body Drop

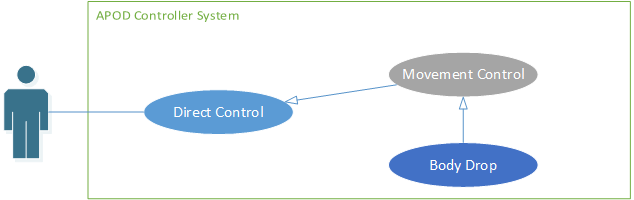


Figure C‑13 Body Drop Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **BODY DROP USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC012 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Body Drop | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Command APOD to Drop down  **Goal:**  APOD Drop body Down  **Triggers**  User click [R2] button  **Preconditions:**   * Live control feature is selected * Bluetooth connection is established * Normal input mode or Gamepad input mode is selected   **Post Conditions:**  NONE  **Main Success Scenario:**   * User click on [R1] button ()   **Alternative Scenario:**   * Alt 1: press “O” key on keyboard (if Normal input is selected) * Alt 2: press () key on gamepad (if Gamepad input is selected)   **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑12

###### Squeeze Left

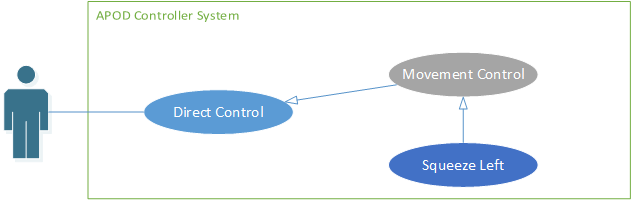


Figure C‑14 Squeeze Left Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SQUEEZE LEFT USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC013 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Squeeze Left | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Command APOD to squeeze to the left  **Goal:**  APOD Drop Squeeze left  **Triggers**  User click [Turn Left] button  **Preconditions:**   * Live control feature is selected * Bluetooth connection is established * Normal input mode or Gamepad input mode is selected * [L2] button is pressed   **Post Conditions:**  NONE  **Main Success Scenario:**   * User press [L2] button () * User click on [Turn Left] button ()   **Alternative Scenario:**   * Alt 1: press “A” key on keyboard while pressing “E” key (if Normal input is selected) * Alt 2: press () key on gamepad while pressing  (if Gamepad input is selected)   **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑13

###### Squeeze Right

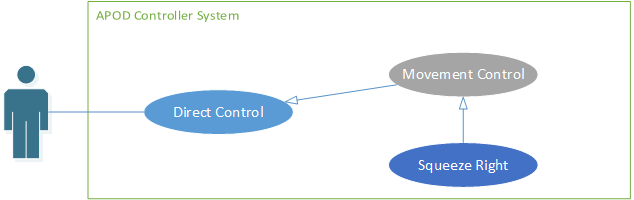


Figure C‑15 Squeeze Right Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SQUEEZE RIGHT USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC014 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Squeeze Right | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Command APOD to squeeze to the right  **Goal:**  APOD Drop Squeeze right  **Triggers**  User click [Turn Right] button  **Preconditions:**   * Live control feature is selected * Bluetooth connection is established * Normal input mode or Gamepad input mode is selected * [L2] button is pressed   **Post Conditions:**  NONE  **Main Success Scenario:**   * User press [L2] button () * User click on [Turn Left] button ()   **Alternative Scenario:**   * Alt 1: press “D” key on keyboard while pressing “E” key (if Normal input is selected) * Alt 2: press () key on gamepad while pressing  (if Gamepad input is selected)   **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑14

###### Manual Grip

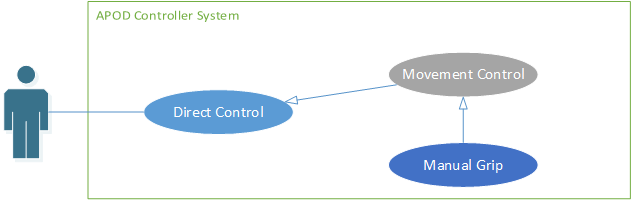


Figure C‑16 Manual Grip Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MANUAL GRIP USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC015 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Manual Grip | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Command APOD to Grip Mandibles  **Goal:**  APOD Mandibles Grip  **Triggers**  User click [Grip] button  **Preconditions:**   * Live control feature is selected * Bluetooth connection is established * Normal input mode or Gamepad input mode is selected * [L1] and [L2] is not pressed.   **Post Conditions:**   * APOD stop when button is released   **Main Success Scenario:**   * User click on [Grip] button ()   **Alternative Scenario:**   * Alt 1: press “I” key on keyboard (if Normal input is selected) * Alt 2: press () key on gamepad (if Gamepad input is selected)   **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑15

#### Object tracking

##### Auto Grip

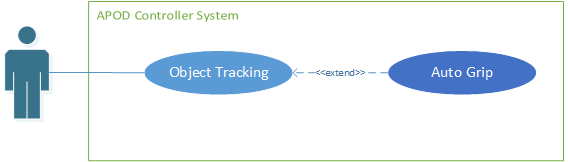


Figure C‑17 Auto Grip Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **AUTO GRIP USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC016 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Auto Grip | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Auto detect object in range and move forward to grab it.  **Goal:**  Object is grabbed  **Triggers**  User click [AUTO GRIP] button  **Preconditions:**   * Live control feature is selected * Bluetooth connection is established   **Post Conditions:**  NONE  **Main Success Scenario:**   * User click [AUTO GRIP] button * Auto grip command is sent to APOD   **Alternative Scenario:**  NONE  **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑16

##### Choose Target

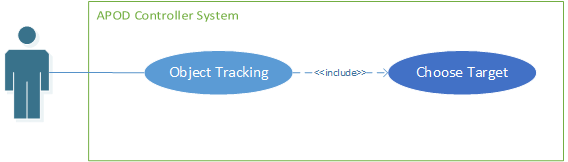


Figure C‑18 Choose Target Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CHOOSE TARGET USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC017 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Choose Target | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Enter tracking mode, choosing target to track  **Goal:**  New tracking template is selected  **Triggers**  User click [Object Tracking] button  **Preconditions:**   * Configuration is initiated * Bluetooth connection is established * IP camera stream is working   **Post Conditions:**  NONE  **Main Success Scenario:**   * [Object Tracking] button is clicked * Object extractor is showed * User select tracking template from capture frame. * User click [OK]   **Alternative Scenario:**  NONE  **Exceptions:**   * IO Exception * Time Out Exception   **Relationships:**  NONE | | | | |

Table C‑17

##### Change Target

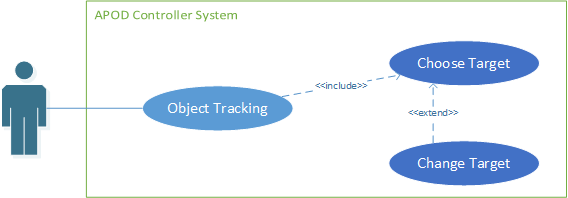


Figure C‑19 Change Target Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ADD POINT USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC018 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Change Target | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Change current tracking template  **Goal:**  New tracking template is selected  **Triggers**  User click [Change target] button  **Preconditions:**   * Configuration is initiated * Bluetooth connection is established * IP camera stream is working * Tracking mode is selected   **Post Conditions:**  NONE  **Main Success Scenario:**   * [Change Target] button is clicked * Object extractor is showed * User select tracking template from capture frame. * User click [OK]   **Alternative Scenario:**  NONE  **Exceptions:**   * IO Exception * Time Out Exception   **Relationships:**  NONE | | | | |

Table C‑18

##### Lock Target

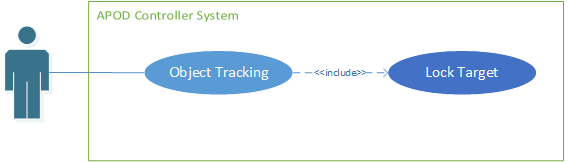


Figure C‑20 Lock Target Use-case diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ADD POINT USE CASE SPECIFICATION** | | | | |
| **Use-case No.** | UC019 | **Use-case Version** | | 1.0 |
| **Use-case Name** | Lock Target | | | |
| **Author** | CuongPAD | | | |
| **Date** | 05/10/2013 | **Priority** | Normal | |
| **Actor:**  User  **Summary:**  Start tracking target  **Goal:**  Reach target (within 60 cm)  **Triggers**  User click [Lock] button  **Preconditions:**   * Configuration is initiated * Bluetooth connection is established * IP camera stream is working * Tracking template was set   **Post Conditions:**  NONE  **Main Success Scenario:**   * User click on [Lock] button * APOD keep moving forward (with adjustment) until target is reached * User decide to Grab object or not.   **Alternative Scenario:**  NONE  **Exceptions:**   * IO Exception   **Relationships:**  NONE | | | | |

Table C‑19

## SOFTWARE SYSTEM ATTRIBUTES

### Reliability

* Easy to upgrade firmware.
* The hexapod controller can be replaced easily by loaded into chip if the controller has problem.
* The APOD system is guarantee by quality testing in:
  + Stability constraints.
  + Functionality.
  + Reliability.
* It’s mostly depending on hardware reliability.
* Small error margin when moving.

### Availability

* In case of electrical incident, the hexapod system will be shut down and reset automatically.
* Controller has the flexibility that allows changes in hardware design.
* Hexapod Controller can actuate the 6-legs forces to properly position the mobile plate given a desired trajectory.

### Security

* N/A

### Maintainability

* N/A

### Portability

* The hexapod controller is depend on hardware so that hexapod system do not have the portability attribute. However, the hexapod controller can easily be loaded into chips and use in different motors.
* The development environment and the language constructs used ensure portability as much as possible to avoid the limitation of software’s changes each time the hardware is upgraded or replaced.

### Performance

* Hexapod controller can control multi-motors in the same time.
* The current hardware can move correctly

## ENTITY RELATIONSHIP DIAGRAM

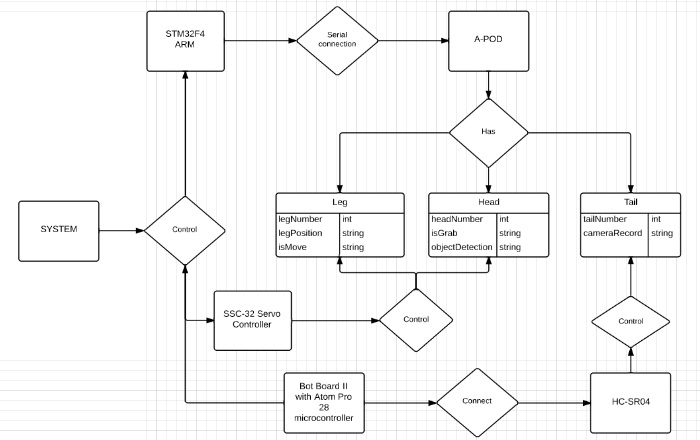


Figure C‑21

# SOFTWARE DESIGN DESCRIPTION (SDD)

## DESIGN OVERVIEW

### Introduction

This section is created to introduce and give a brief overview of the system design.

The followings information are given in this section:

* System Architecture Design : the system overall structure
* Components Description : detailed info of hardware components
* Class Diagram : entity & attributes
* Sequence Diagram : describe the flow of events
* User Interface Design

### Architecture

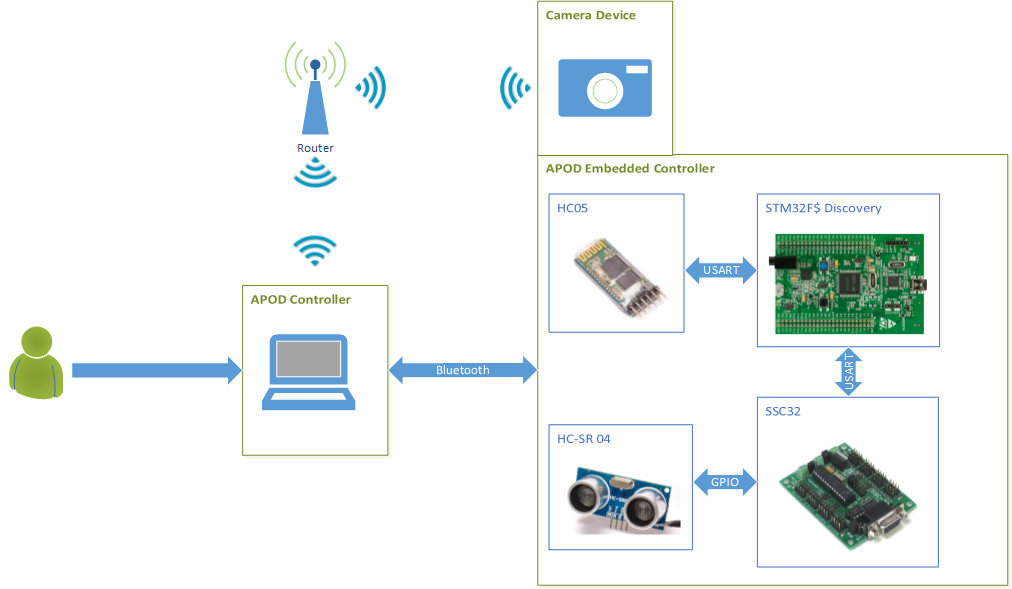


Figure D‑1 APOD Controller System Architecture

### User interface

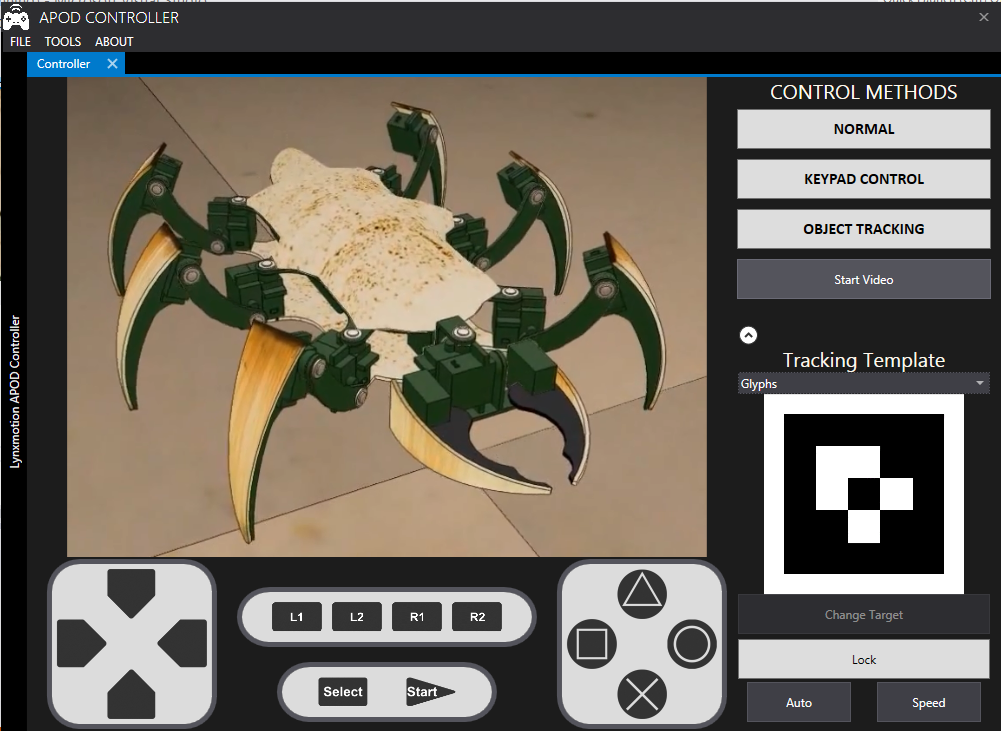


Figure D‑2 Live Control User interface design

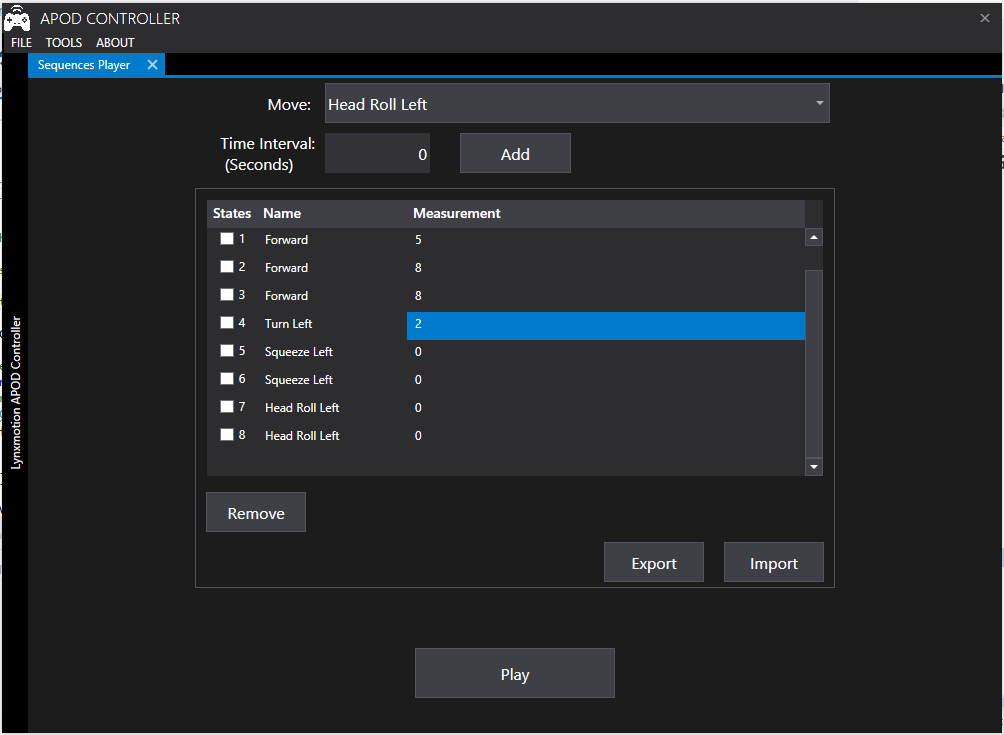


Figure D‑3 Sequence Player user interface design

## COMPONENTS

### Central processing:

#### Bluetooth module (HC 05)

Communication module (using Bluetooth) to transmit and receive commands from Controller devices (PC)

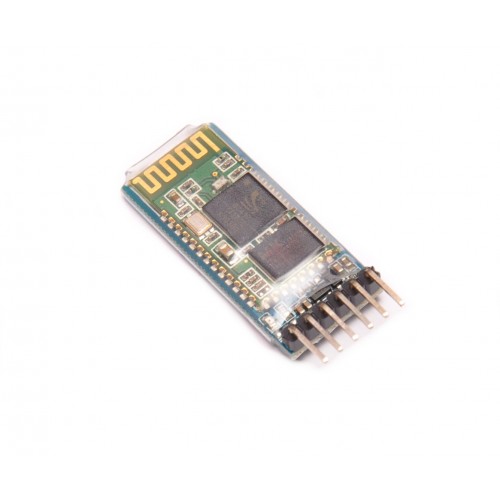


Figure D‑4 Bluetooth Communication module HC05

#### STM32F4 Discovery

The Center Processing module to control all APOD movements or reaction to a specific event (command or obstacle).

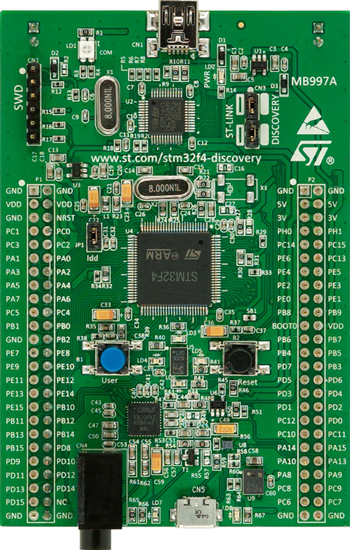


Figure D‑5 STM32F4 Discovery Board

### Servos control & Sensor reader

#### HG-645MG Servo

* Metal gear servo that provided the base of APOD movement.



Figure D‑6 HC 645MG Servo

#### SSC32 with ATMEGA328P

* A servo controller : contains 32 pin channels of 1uS resolution servo control
* For further information, followings this link : [Lynxmotion SSC32](http://www.lynxmotion.com/images/html/build136.htm)

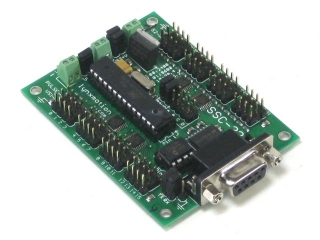


Figure D‑7 SSC32 Board

#### Ultrasonic ranging module (HC SR04)

* Ultrasonic module using for distance measurement, discovering obstacles.

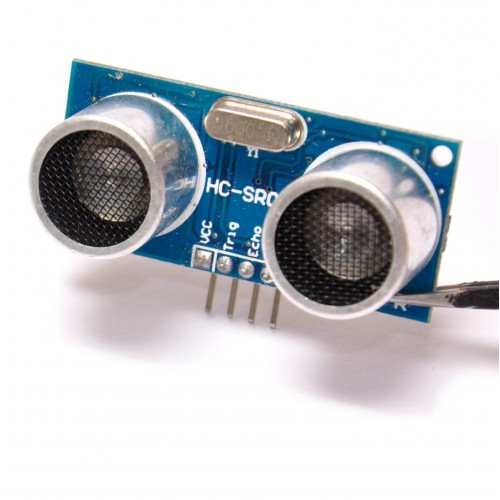


Figure D‑8 HCSR04 Distance Sensor

## IMPLEMENTATION APPROACHES

### Algorithms and Mechanism

#### RC Servo Control

##### HG-645MG Specification

|  |  |
| --- | --- |
| Motor Type: | 3 Pole |
| Bearing Type: | Dual Ball Bearing |
| Speed (4.8V/6.0V): | 0.24 / 0.20 |
| Torque oz./in. (4.8V/6.0V): | 107 / 133 |
| Torque kg./cm. (4.8V/6.0V): | 8.0 / 10.0 |
| Size in Inches: | 1.59 x 0.77 x 1.48 |
| Size in Millimeters: | 40.39 x 19.56 x 37.59 |
| Weight ounces: | 1.94 |
| Weight grams: | 55 |

##### How does it work?

Unlike the AC/DC motor, Servo motor do not rotate continuously. Servo motor rotation range usually fall within 90o to 180o (some special servo can rotate up to 360o) and keep holding the torque at the position as long as its power supply remain or until it receive signal to move the torque.

Most of Servos have 3 wires connector. One wire for power supply (4 – 6v). The second wire is for voltage ground. And the third wire is for receiving control signal.

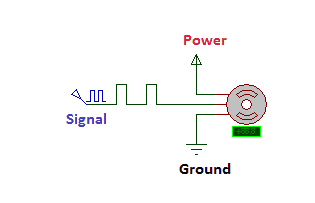


Figure D‑9 RC Servo block

The whole idea to control servo is sending the appropriate signal to servo so that it will move to the intended position. The signal to control servo, basically is a PWM (Pulse Width Modulation) signal of 50 Hz (or refresh rate: 20ms). It is the width of carrier pulse that decide the position of the torque (not the number of pulse).

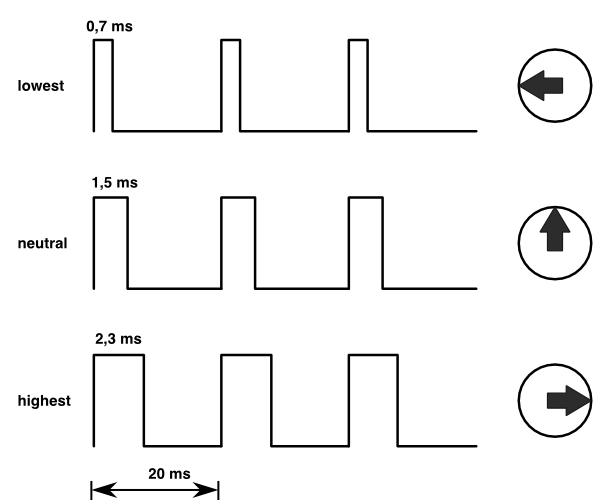


Figure D‑10 Servo Signal illustration

In order for the servo to keep the position, the signal must be maintain.

In short words, the pulse width of carrier signal will act as a coordinate value of the servo’s torque. Once the coordinate is changed, the servo constantly move the toque to the responding position discarding of its current position. The rotating speed will be depending on the manufacturer of the servo.

For the HG-645MG servo, the corresponding signal is showed below:

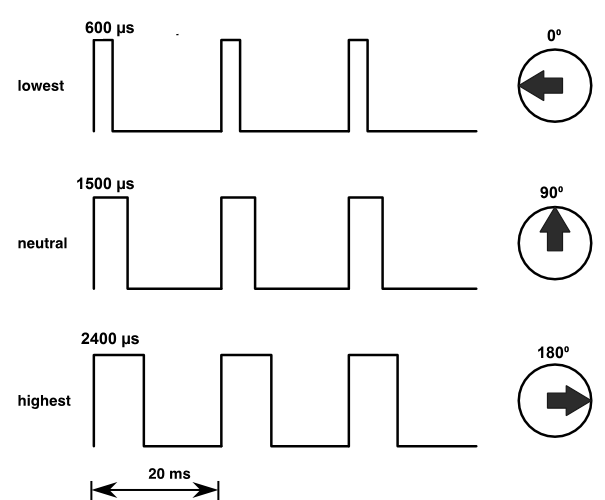


Figure D‑11 HC 645MG Signal Details

##### Servo’s Speed Control

As we know, the servo will always rotate at a constant speed, for example the servo HG-645MG speed is 60o/ 0.2s with 6V supply. With such speed applied to robot’s movement, it can affect the stability of the structure, especially for those in which mechanical part is not so much reliable. How do we slow it down?

The idea is not making the servo rotate a whole distance at once but make it rotate one step at a time until it reach its destination, in short it is “divide and conquer” method. For example, you want your HG-645MG servo to rotate 90o in 1 second and the current position of your servo is 0o. If you just apply the PWM with 1500 µs pulse width (1500 µs stand for 90o position), the servo will move to its destination in only 0.3 second. With the method mentioned above, we divide the whole 90o into ten 9o pieces (or 150 µs step as we call it). The signal pulse width will be modified 10 times, after each time the width will be added 150 µs more and waiting 100 ms until the next change happen. With this implementation, the servo will move 10x9o = 90o in 10x100 = 1000 ms = 1 s. The larger the divisor is, the smoother the servo will move.

For this project, the step is always fix at 20 µs and the waiting time fall in range of 10 ms to 20 ms depend on user purposes.

Let D be the travelling distance (in µs), T be the expected time for the move, δ be the waiting time for each step. We have the speed formula:

#### Sensor Reader (HC-SR04)

##### HC-SR04 Specifications

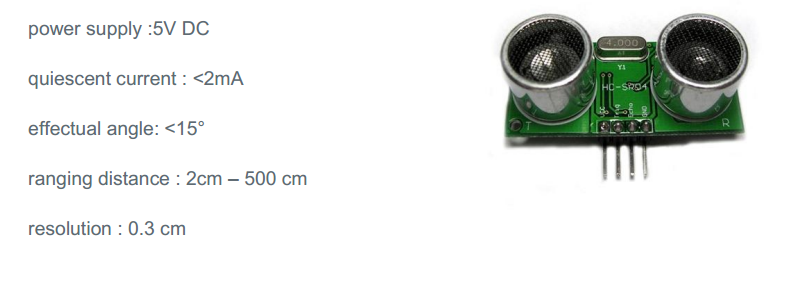


Figure D‑12 HC-SR04 Specifications

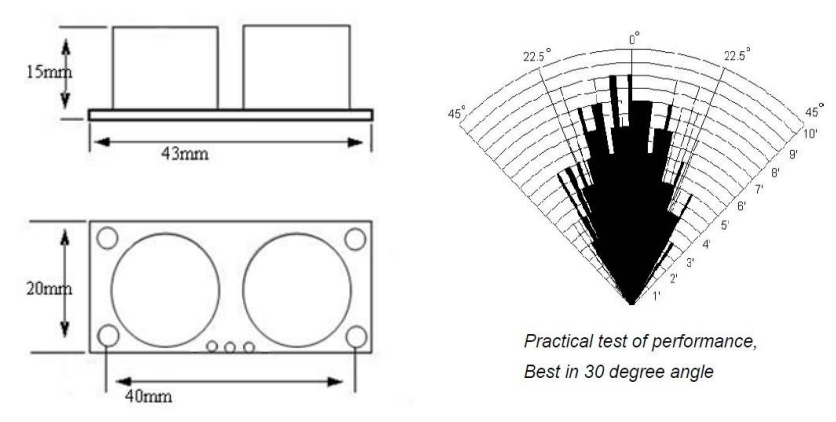


Figure D‑13 HC-SR04 Best practice angle

##### How to read distance value?

The HC-SR04 has 4 pins: VCC, GND, Trigger, Echo.

* VCC is used for power supply
* GND is used for voltage ground
* Trigger and Echo will perform as follow

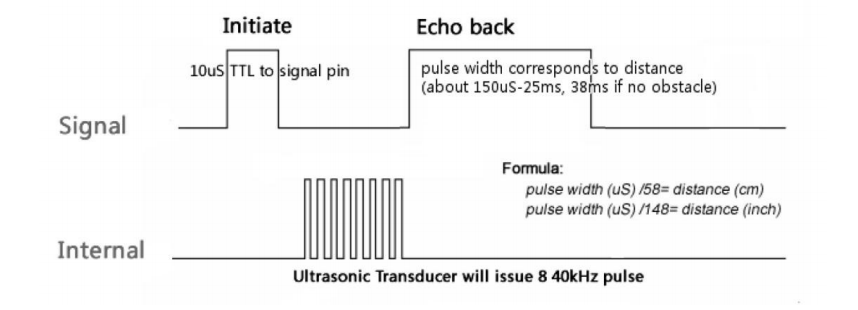


Figure D‑14 HC-SR04 Operating sequence

First the Trigger pin is pull high for more than 10 µs (10 µs pulse), then waiting for the Echo output to go high and start timing the length of the Echo pin staying high. At this point, a short ultrasonic pulse is transmitted, waiting to be reflected by an obstacle. When the reflected ultrasonic pulse is receive by senor, the Echo output will go low.

The time that Echo output staying high will be used to calculate the distance by following formula given by the manufacturer:

Or

#### Multi-legs Control

The description below is described in the following article by Oricom Technologies– “Analysis of Multi-Legged Animal + Robot Gaits” (2003).

##### Tripod Gait

The tripod gait is the best-known hexapod gait. Hexapod six legs is divide into 2 halves for to control. Each half (or tripod) consists of 3 legs: the front and rear leg of one side and the center leg of the opposite side. More specific:

* Tripod A: left front, right center and left rear
* Tripod B: right front, left center and right rear.

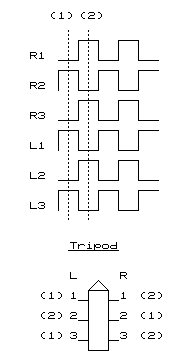


Figure D‑15 Tripod Gait illustration

During walking cycle, the weight of the entire hexapod is shifted from one tripod to another. At a moment, there is always one tripod on the ground while another doing its cycle. “*Since 3 legs are on the ground at all times, this gait is both "statically" and "dynamically" stable*.” - (Oricom Technologies , 2003)

##### Wave Gait

The wave gait implementation make each leg in one side do it cycle after one another leg has finished its cycles and repeat it on the other side’s legs.

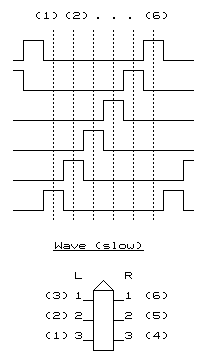


Figure D‑16 Wave Gait illustration

For instance, the turn of leg taking step is as follow: Left Front -> Left Center -> Left Rear -> Right Front -> Right Center -> Right Rear

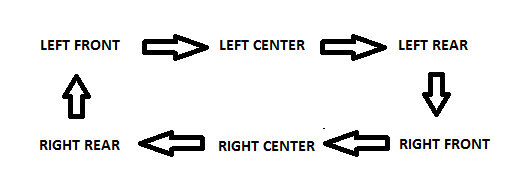


Figure D‑17 Wave Gait step order

Since there is always 5 legs on the ground, this method is the most stable gait but in return, it is also the slowest gait due to its “*one step at a time*”.

##### Ripple Gait

“*The final stride is the Ripple Gait. At first glance the timing of this gait looks somewhat complicated, however, the key to understanding is to recognize that, on each side a local wave comprising non-overlapping lift phases is being performed, and that the 2 opposite side waves are exactly 180 degrees out of phase with one another. For instance, if L3 and R3 are considered to represent the start of each local wave, then notice that R3 starts to move exactly in the middle of the L3-L2-L1 side wave*” .” (Oricom Technologies , 2003)

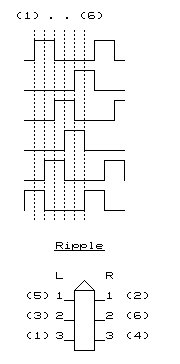


Figure D‑18 Ripple Gait illustration

##### Implementation Approaches

For this project, we will apply **the tripod gait** method to control the APOD walking movement. As we know, 6 legs of APOD will be divided into 2 tripods

We defined 2 tripod as follow:

* **Tripod A: Left front, Right Center, Left Rear**
* **Tripod B: Right front, Left Center, Right Rear**

###### Walking In Straight Line

**Leg’s Position:**

There are 6 basic positions for the APOD’s leg divive into 2 orientation.

* Vertical: High, Middle , Low
* Horizontal: Front, Center, Rear

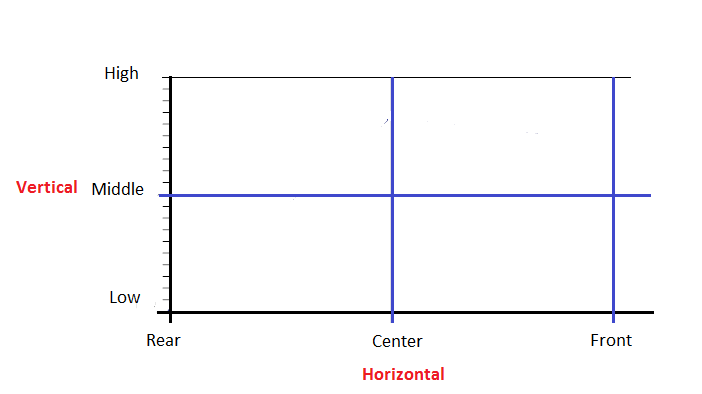


Figure D‑19 APOD leg position block

Each step (swing time) of the APOD leg is represented by a sequence of 4 consecutive states defined by the position of the leg in both team of vertical and horizontal positions.

For example, this is the forward step sequence:

|  |  |  |
| --- | --- | --- |
| State | Vertical | Horizontal |
| 0 | Low to Mid | Rear |
| 1 | Mid to High | Rear to Center |
| 2 | High to Mid | Center to Front |
| 3 | Mid to Low | Front |

Table D‑1

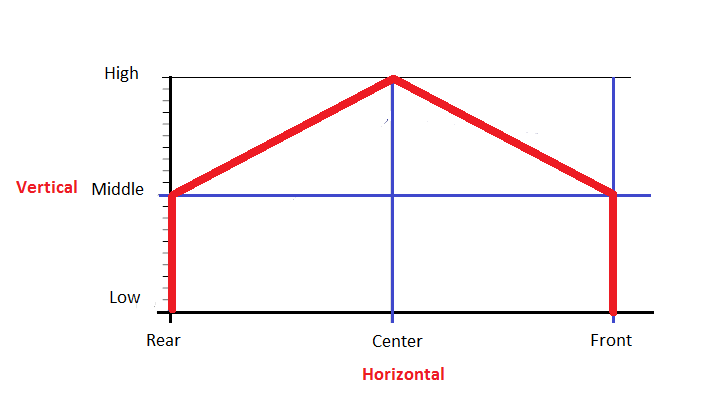


Figure D‑20 APOD leg swing diagram

Depend on the step direction (forward or backward) the sequence may differ: “Front” and “Rear” may be replaing each other’s position.

After making the step, the leg will return to its original position (compare to the body) while keeping the feet on the ground (stand time). This is the point the force was born to push the body moving in the opposite direction of the moving legs.

From the APOD view point, the whole operations of moving leg can be seen as the figure below:

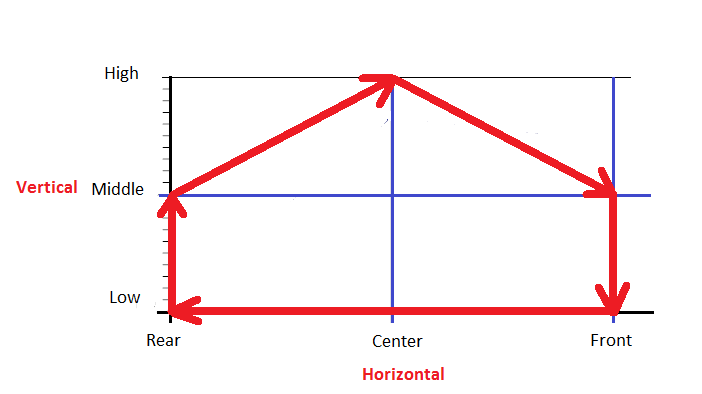


Figure D‑21 APOD Leg full step diagram

The whole operation of APOD’s leg can be seen as this sequence of states (Forward step) looping times to times:

|  |  |  |
| --- | --- | --- |
| **State** | **Vertical** | **Horizontal** |
| 0 | Low | Front to Center |
| 1 | Low | Center to Rear |
| 2 | Low | Rear |
| 3 | Low to Mid | Rear |
| 4 | Mid to High | Rear to Center |
| 5 | High to Mid | Center to Front |
| 6 | Mid to Low | Front |
| 7 | Low | Front |

Table D‑2 APOD Leg full forward step

**Tripod Position**

According to the nature of The Tripod Gait, every 3 legs in one tripod will always move and stop at the same time. In other words, their cycles are exactly the same phase with one another. The term tripod position is defined by the combination of its legs positions.

Let’s take a view of Tripod A (marked with red color) from the up top viewpoint for example:

* Position: Front (close to head)

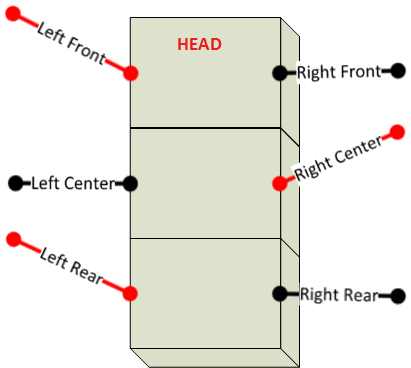


Figure D‑22 Tripod A at Front position

* Position: Center or Neutral

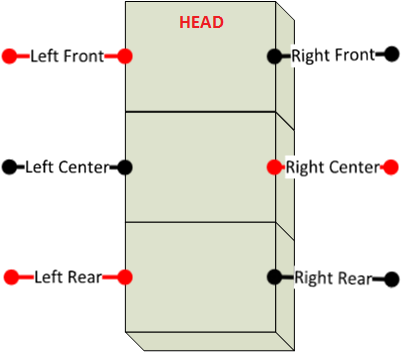


Figure D‑23 Tripod A at Neutral position

* Position: Rear (far from head)

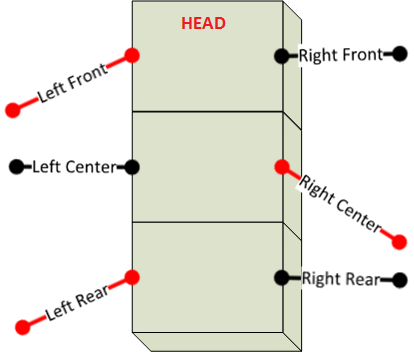


Figure D‑24 Tripod A at Rear position

Since all Tripod A’s Legs making the forward step at once (moving from front to rear), there will be 2 pushing force from both side of the body, from the back to the head, that making the APOD moving forward. This is the fundamental of making the APOD moving forward. The backward move can be achieve by making the reverse step of forward step (the backward step).

**Walking**

Now before we can figure out the walking method of APOD, let’s examine the human walking method first. As we can see, the position of one leg in walking (or running) is always the opposite position of the other one. Or in short, their cycles is exactly 180o out of phase with one another. The similar mechanism is applied for The Tripod Gait walking method. Let Tripod A and B stand for 2 legs of human, then their cycles will always be 180o out of phase with the other one. In conclusion, the full sequence for 2 tripods of The Tripod Gait walking method is described as below

**Forward Sequence**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **State** | **Tripod A** | | **Tripod B** | |
| **Vertical** | **Horizontal** | **Vertical** | **Horizontal** |
| 0 | Low | Front to Center | Mid to High | Rear to Center |
| 1 | Low | Center to Rear | High to Mid | Center to Front |
| 2 | Low | Rear | Mid to Low | Front |
| 3 | Low to Mid | Rear | Low | Front |
| 4 | Mid to High | Rear to Center | Low | Front to Center |
| 5 | High to Mid | Center to Front | Low | Center to Rear |
| 6 | Mid to Low | Front | Low | Rear |
| 7 | Low | Front | Low to Mid | Rear |

Table D‑3 Forward Sequence

**Backward Sequence**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **State** | **Tripod A** | | **Tripod B** | |
| **Vertical** | **Horizontal** | **Vertical** | **Horizontal** |
| 0 | Low | Rear to Center | Mid to High | Front to Center |
| 1 | Low | Center to Front | High to Mid | Center to Rear |
| 2 | Low | Front | Mid to Low | Rear |
| 3 | Low to Mid | Front | Low | Rear |
| 4 | Mid to High | Front to Center | Low | Rear to Center |
| 5 | High to Mid | Center to Rear | Low | Center to Front |
| 6 | Mid to Low | Rear | Low | Front |
| 7 | Low | Rear | Low to Mid | Front |

Table D‑4 Backward Sequence

###### Turning Left – Right

As mention above, we already know 3 basic position of tripod: Front, Center, Rear. In this section, there are 2 more position: Left and Right.

The definition of 2 new position can be described by tripod A as follow:

* Position: Left

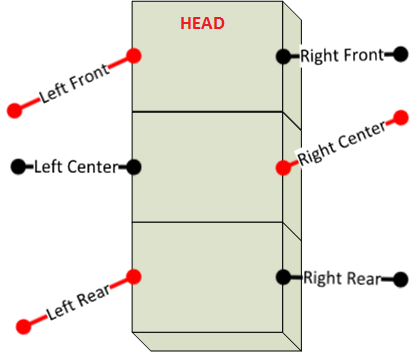


Figure D‑25 Tripod A at Left position

* Position: Right

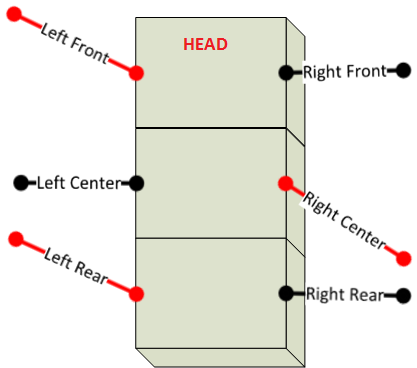


Figure D‑26 Tripod A at Right position

Let’s take a look when Tripod A making set :

* Left Front and Left Rear take the forward step
* Right Center take the backward step

There will be forces with 2 opposite direction on the APOD body. The left side force push the body forward and the right side force push the body backward. As a result of this combination. The APOD will turn from left to right. In other words, when the tripod making transition from left position to right position (through center position), the APOD will turn from left to right. And when the tripod making transition from right position to left position (through center position), the APOD will turn from right to left.

**The combination sequence for turning the APOD:**

**Turn Left:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **State** | **Tripod A** | | **Tripod B** | |
| **Vertical** | **Horizontal** | **Vertical** | **Horizontal** |
| 0 | Low | Left to Center | Mid to High | Right to Center |
| 1 | Low | Center to Right | High to Mid | Center to Left |
| 2 | Low | Right | Mid to Low | Left |
| 3 | Low to Mid | Right | Low | Left |
| 4 | Mid to High | Right to Center | Low | Left to Center |
| 5 | High to Mid | Center to Left | Low | Center to Right |
| 6 | Mid to Low | Left | Low | Right |
| 7 | Low | Left | Low to Mid | Right |

Table D‑5 Turn Left Sequence

**Turn Right:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **State** | **Tripod A** | | **Tripod B** | |
| **Vertical** | **Horizontal** | **Vertical** | **Horizontal** |
| 0 | Low | Right to Center | Mid to High | Left to Center |
| 1 | Low | Center to Left | High to Mid | Center to Right |
| 2 | Low | Left | Mid to Low | Right |
| 3 | Low to Mid | Left | Low | Right |
| 4 | Mid to High | Left to Center | Low | Right to Center |
| 5 | High to Mid | Center to Right | Low | Center to Left |
| 6 | Mid to Low | Right | Low | Left |
| 7 | Low | Right | Low to Mid | Left |

Table D‑6 Turn Right Sequence

### SSC32 Board

#### Schematic

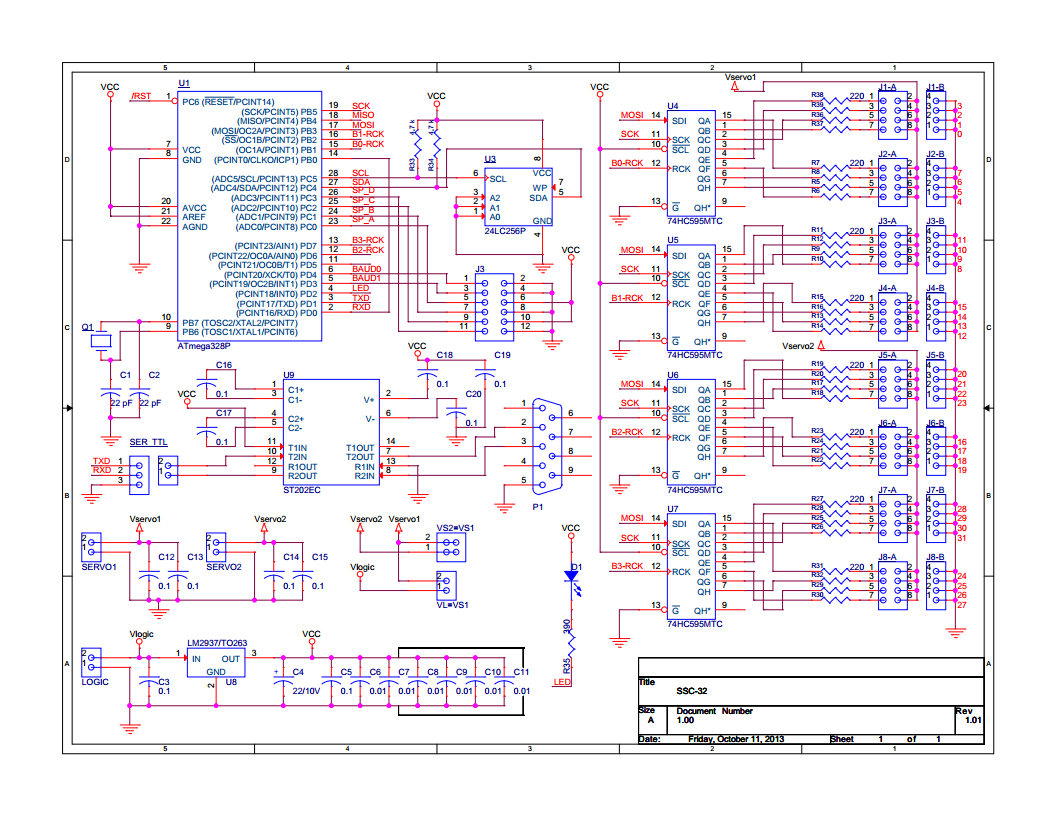


Figure D‑27 SSC32 Schematic

#### Technical Point

##### Atmega328 MCU

##### GPIO

##### USART

##### SPI

##### Shifter (74HC595)

##### SPI – Shifter interface

### STM32F4 Discovery Board

## ACTIVITY AND CLASS STRUCTURE

### Activity

#### APOD Controller (HMI Application)

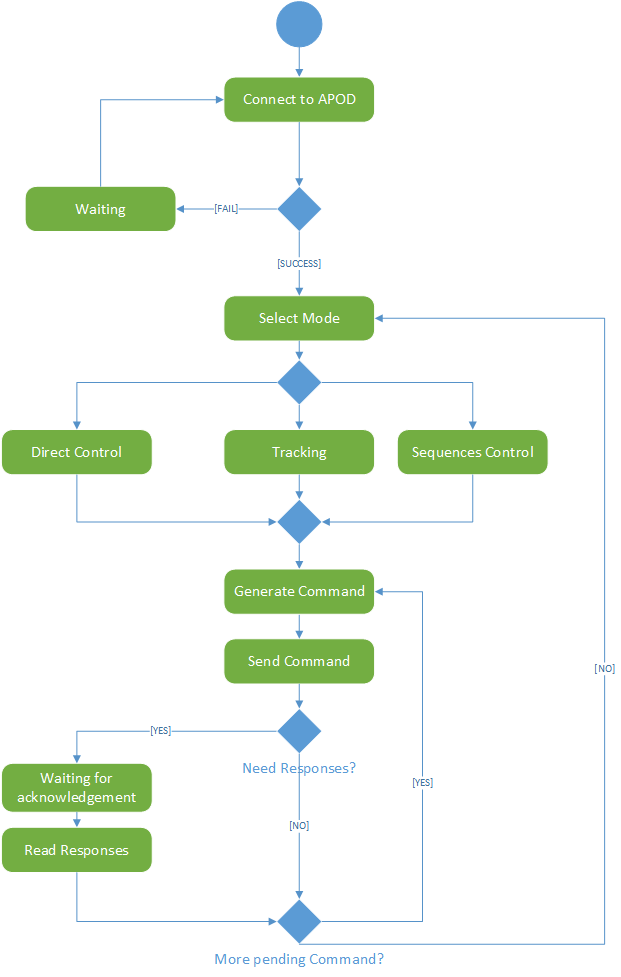


Figure D‑28

#### STM32F4 Discovery (Central processing board)

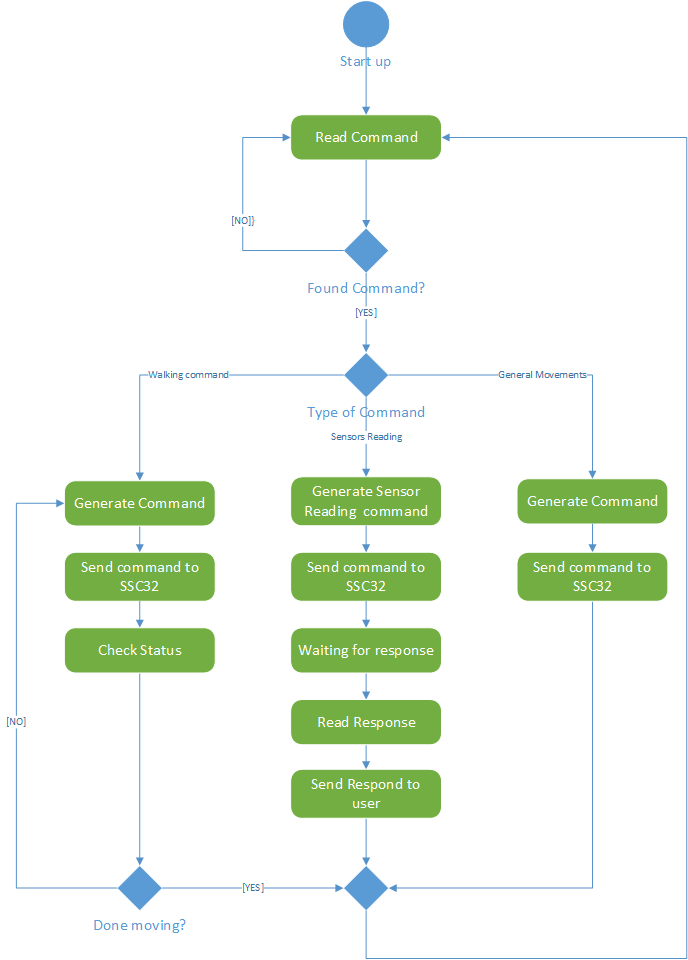


Figure D‑29

#### SSC32 Board (Servos control and Sensor reader)

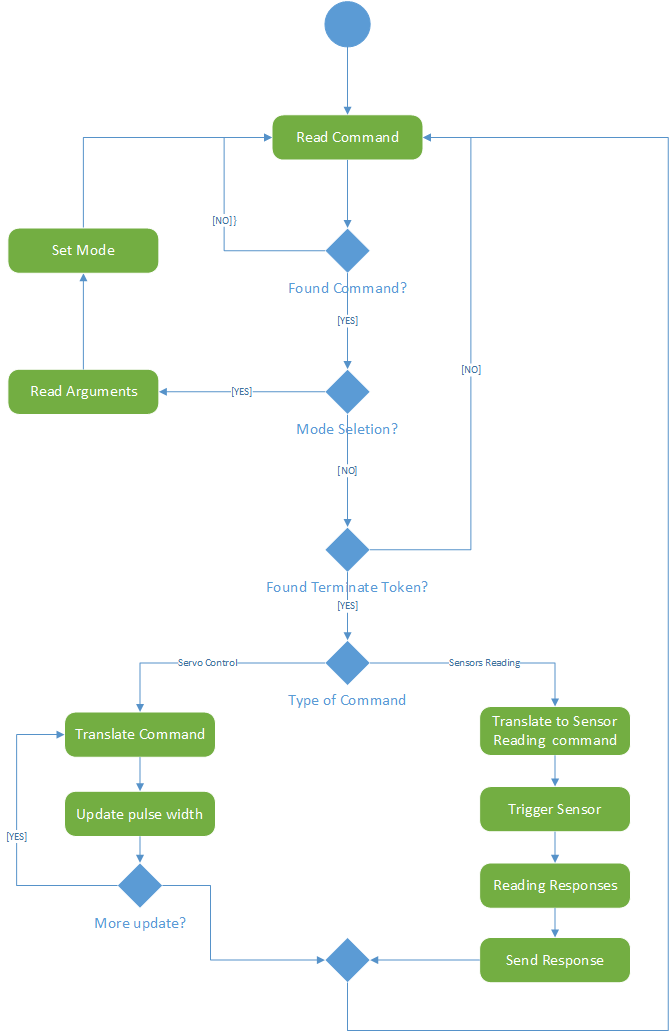


Figure D‑30

### Class Structure

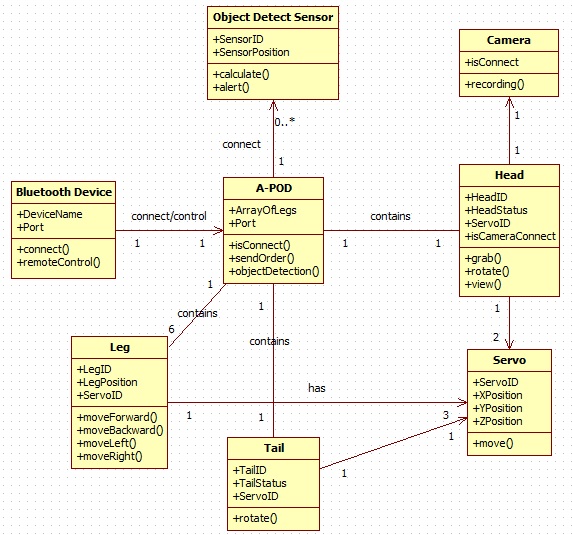


Figure D‑31

## SEQUENCE DIAGRAM

### Embedded Controller

#### Forward

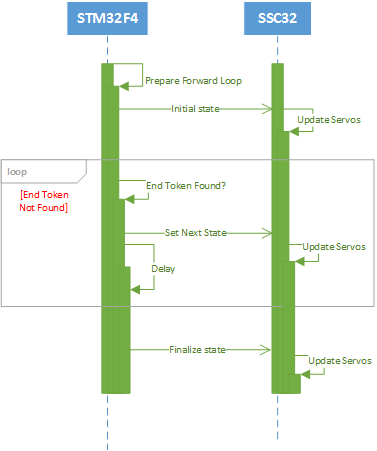


Figure D‑32

#### Backward

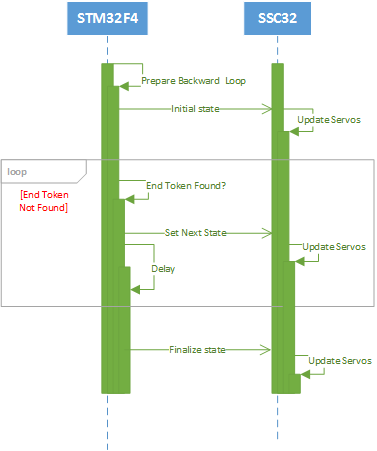


Figure D‑33

#### Turn Left

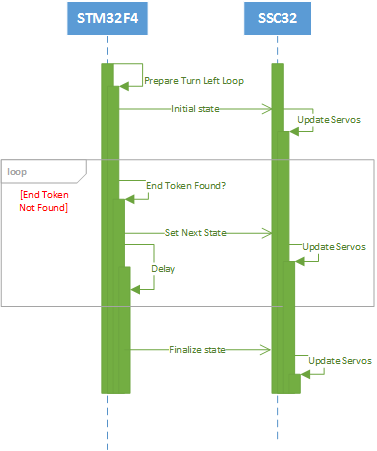


Figure D‑34

#### Turn Right

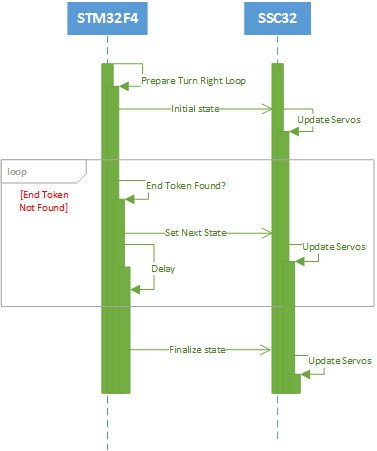


Figure D‑35

### Use-cases Sequences

#### Sequences Control

##### Add State

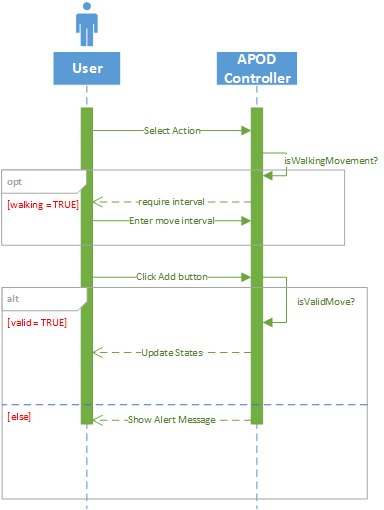


Figure D‑36

##### Remove States

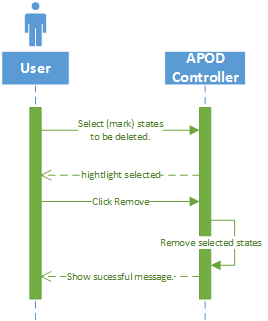


Figure D‑37

##### Import States

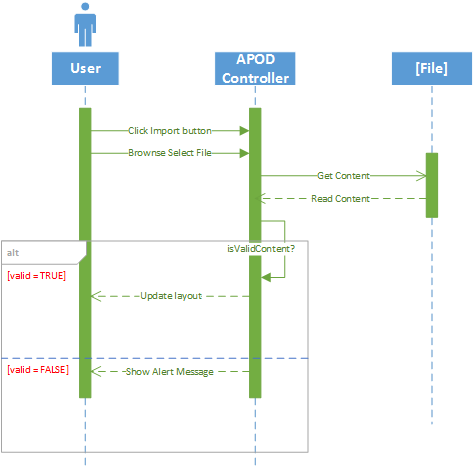


Figure D‑38

##### Export States

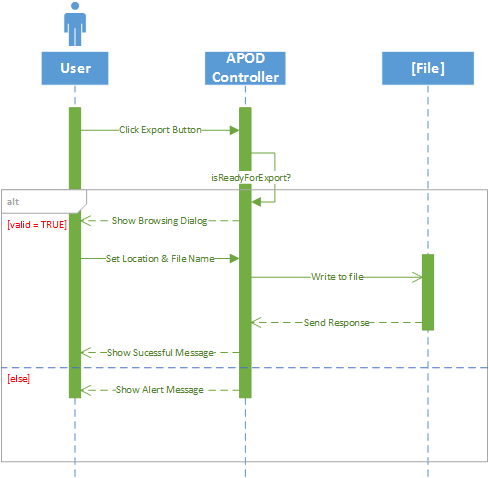


Figure D‑39

#### Direct Control

##### Mode Selection

###### Normal Input

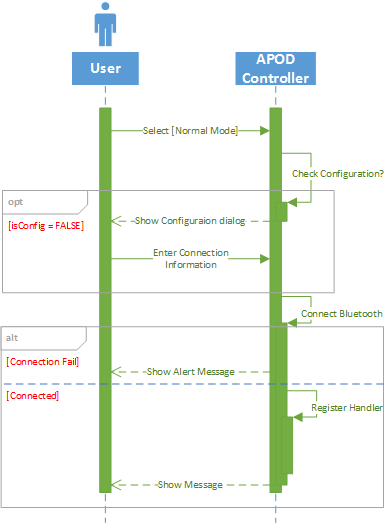


Figure D‑40

###### Gamepad Input

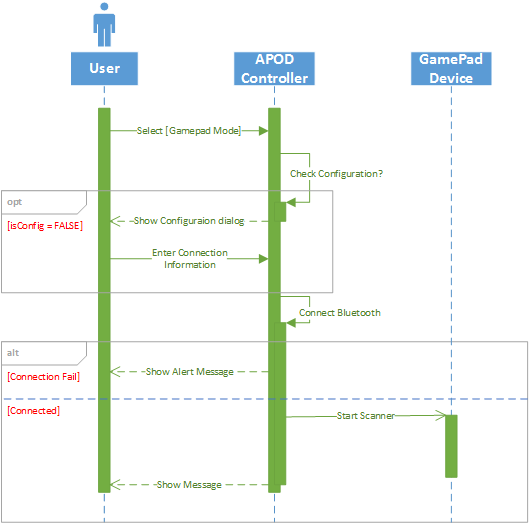


Figure D‑41

##### Movement Control

###### Forward

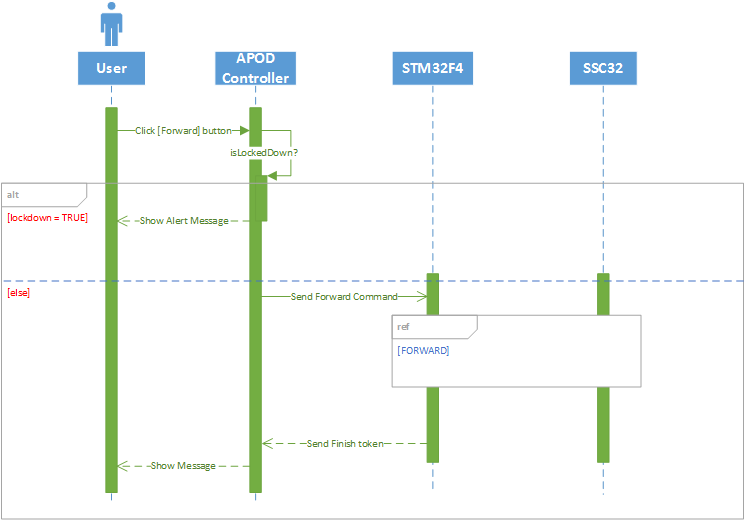


Figure D‑42

###### Backward

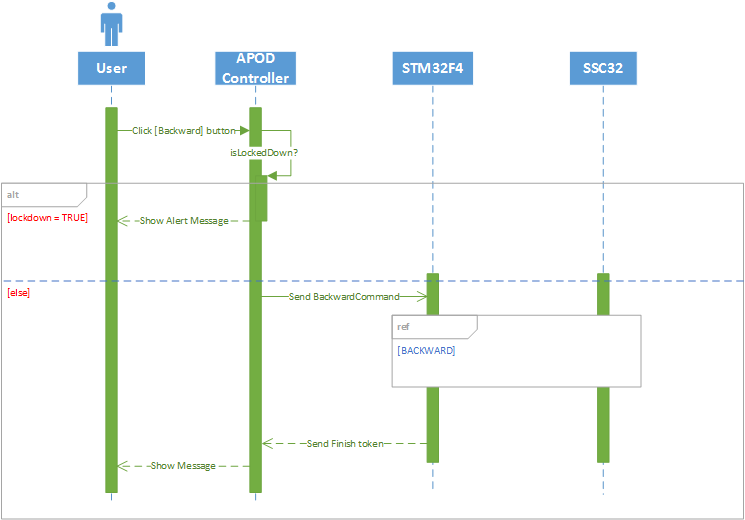


Figure D‑43

###### Turn Left

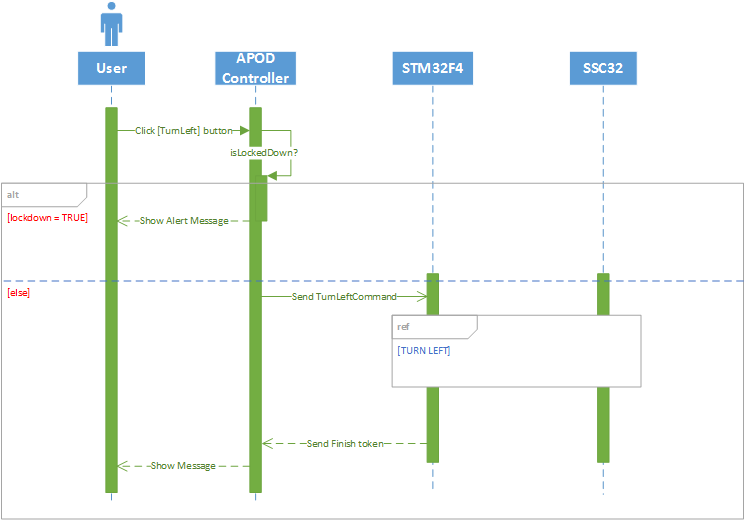


Figure D‑44

###### Turn Right

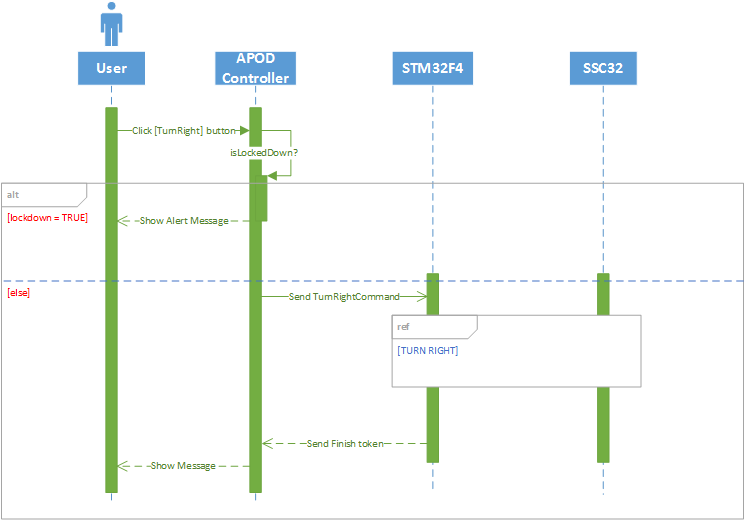


Figure D‑45

###### Body Lift

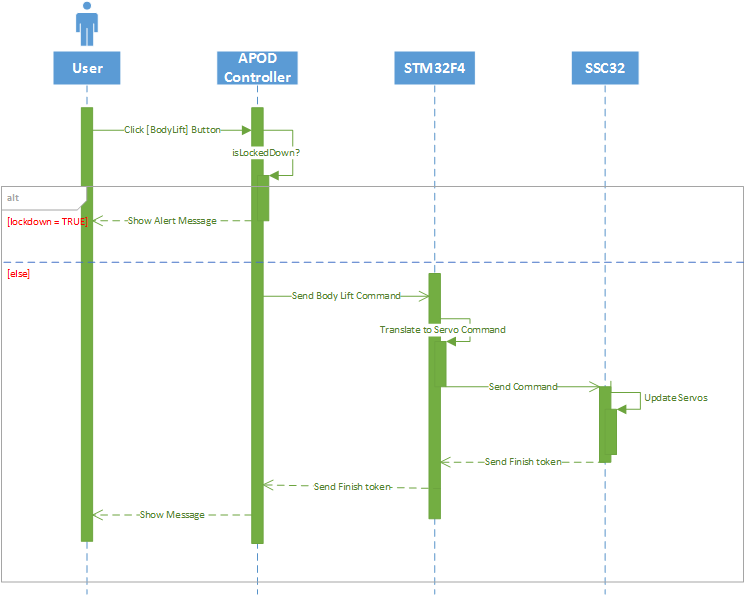


Figure D‑46

###### Body Drop

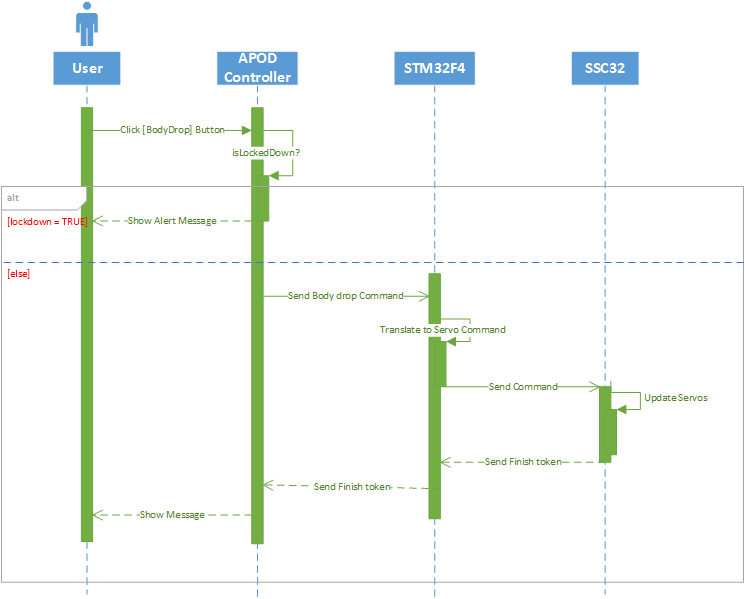


Figure D‑47

###### Squeeze Left

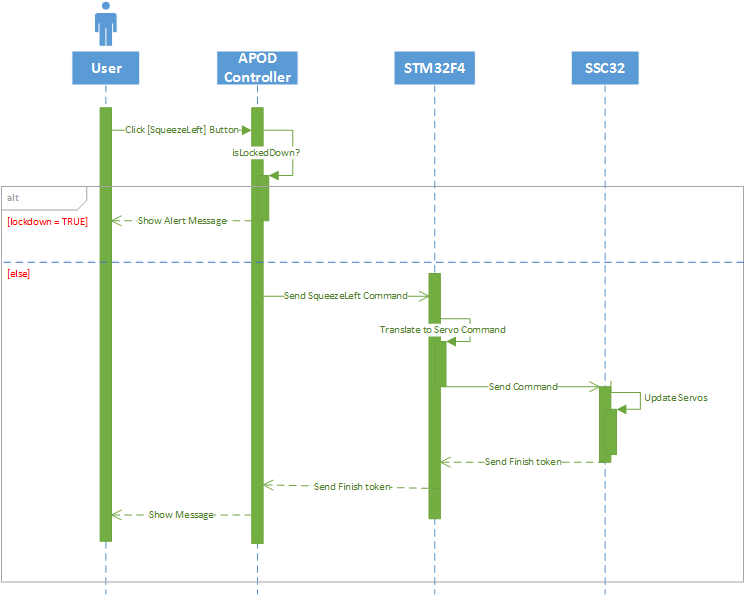


Figure D‑48

###### Squeeze Right

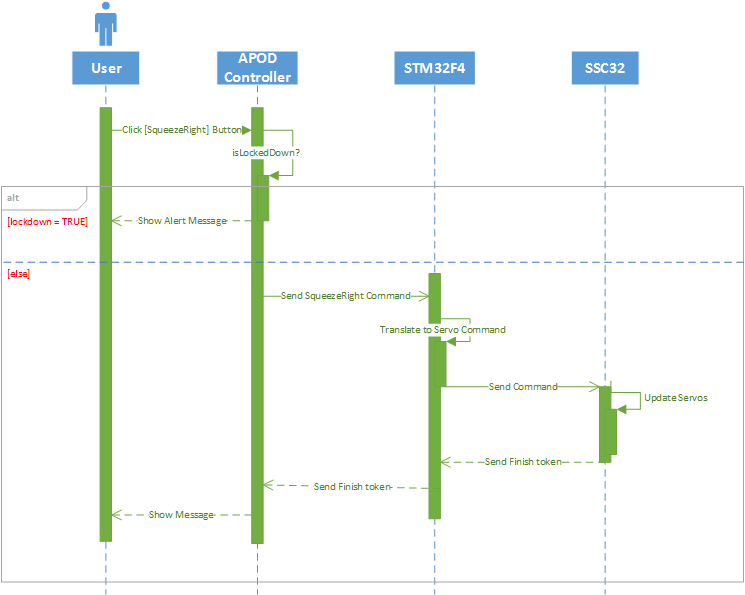


Figure D‑49

#### Object Tracking

##### Auto Grip

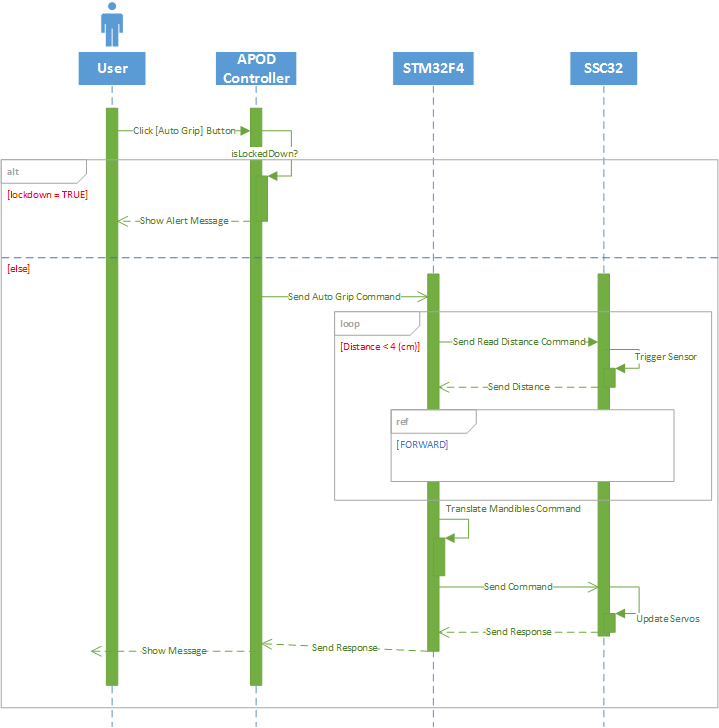


Figure D‑50

##### Choose target

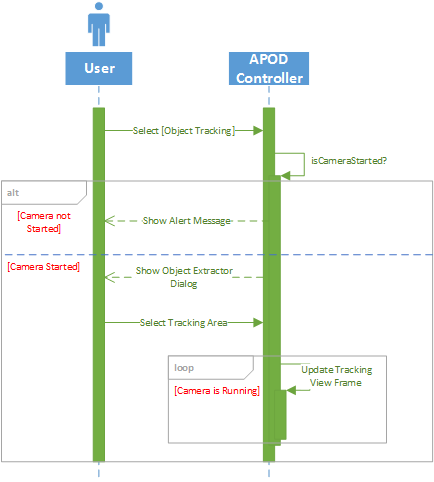


Figure D‑51

##### Change Target

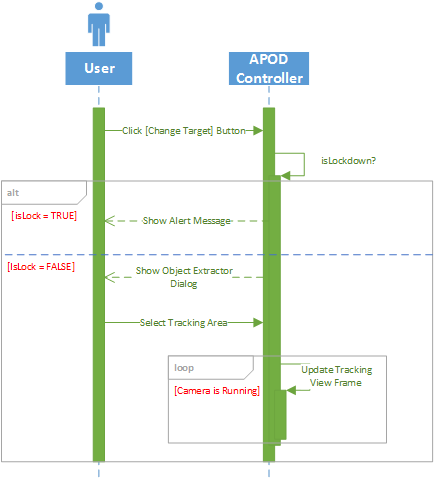


Figure D‑52

##### Lock Target

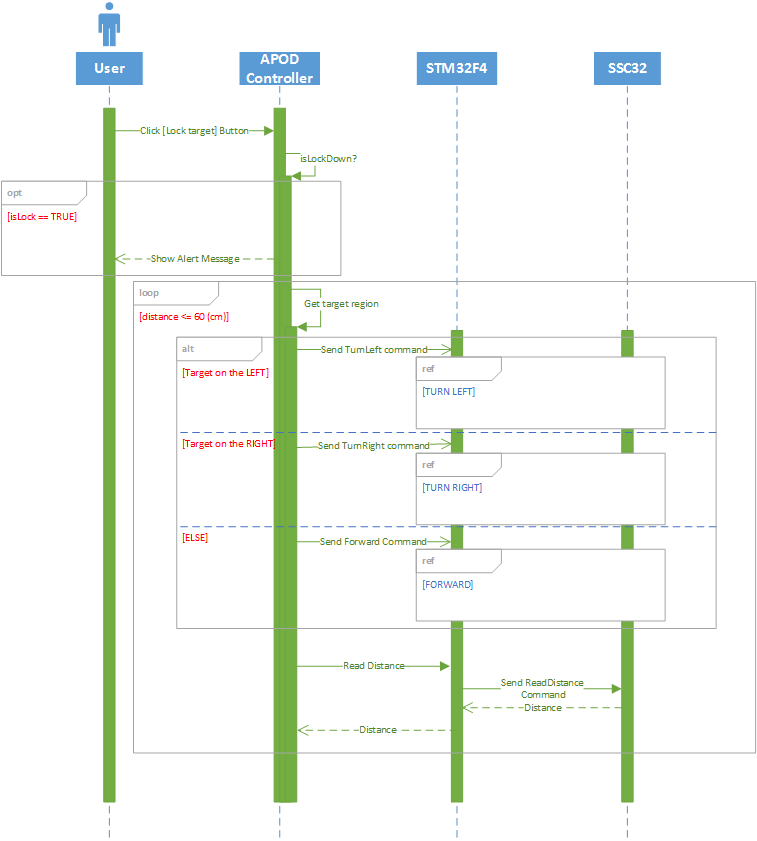


Figure D‑53

## USER INTERFACE DESIGN

### Live Control

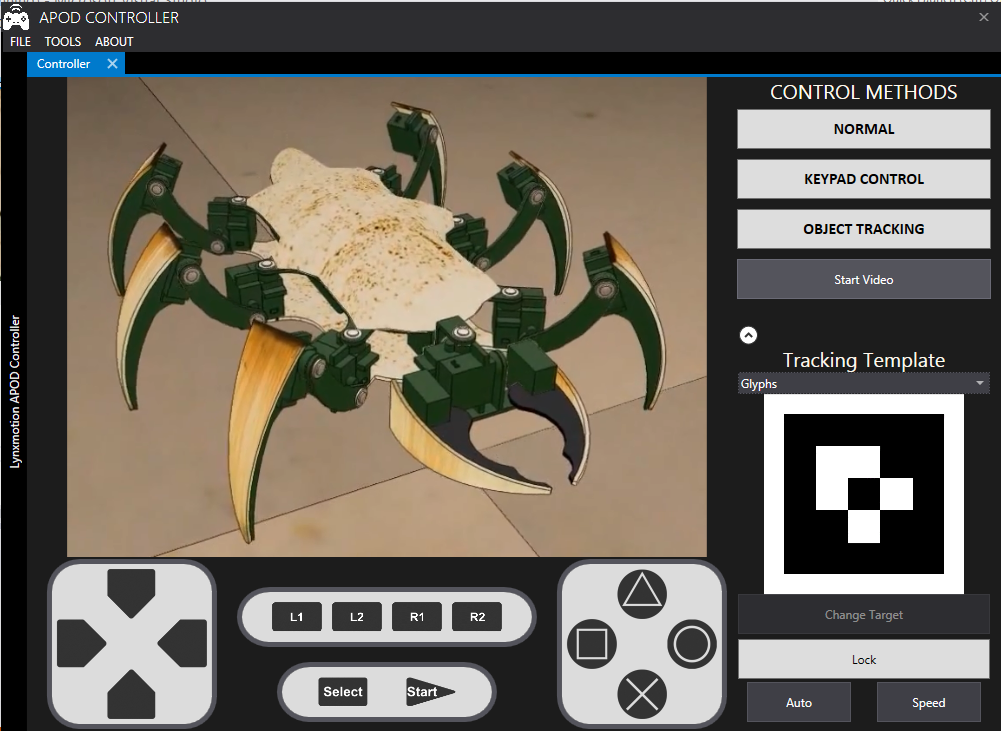


Figure D‑54 Live Control Interface Design

|  |  |  |
| --- | --- | --- |
| Item NO. | Component Name | Description |
| 0 | Menu | Select Features |
| 1 | VideoSourcePlayer | Display Camera IP ‘s View |
| 2 | Checkbox (Button) | Select Normal input mode |
| 3 | Checkbox (Button) | Select Gamepad input mode |
| 4 | Checkbox (Button) | Select object tracking mode |
| 5 | Button | Start streaming video from Camera |
| 6 | Image | Display tracking template |
| 7 | Button | Change Tracking template |
| 8 | Checkbox (Button) | Lock target – Start tracking |
| 9 | Button | Switch speed |
| 10 | Button | Auto grip |
| 11 | Keypad | Action selection |
| 12 | Keypad | Action bank selection |
| 13 | Keypad | Set mode/function |
| 14 | Keypad | Navigation Selection |
|  |  |  |

Table D‑7 Live Control Interface Design Description

### Sequence Player

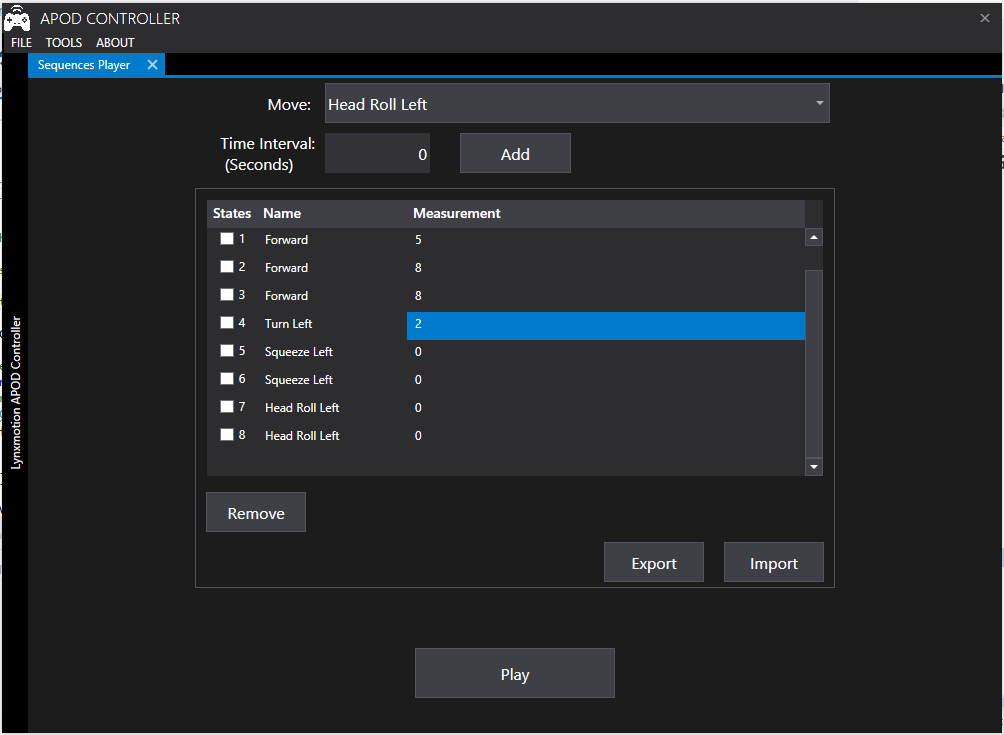


Figure D‑55 Sequence Player Interface Design

|  |  |  |
| --- | --- | --- |
| Item NO. | Component Name | Description |
| 0 | Combo Box | Select action to add |
| 1 | Textbox | Movement interval |
| 2 | Button | Add action to sequence |
| 3 | Data Grid | Display current sequence content |
| 4 | Button | Remove selected states from sequence |
| 5 | Button | Export current sequence to file |
| 6 | Button | Import sequence from file |
| 7 | Button | Start playing the sequence |
|  |  |  |
|  |  |  |
|  |  |  |

Table D‑8 Sequence Player Interface Design Description

### Configuration

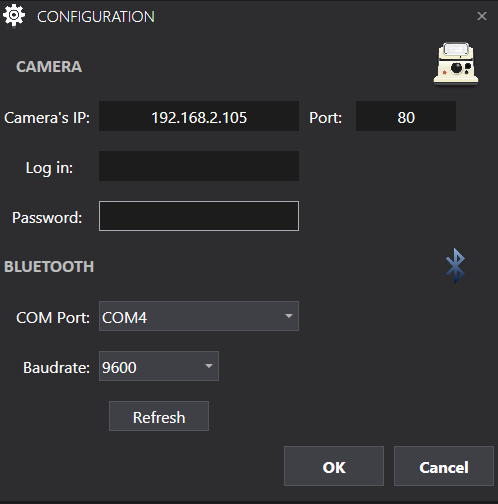


Figure D‑56 Configuration Interface Design

|  |  |  |
| --- | --- | --- |
| Item NO. | Component Name | Description |
| 0 | Textbox | Enter IP of camera |
| 1 | Textbox | Camera port to get stream |
| 2 | Textbox | Username of camera IP |
| 3 | Password Box | Password of camera IP |
| 4 | Combo Box | All available COM ports |
| 5 | Combo Box | Bluetooth baudrate selection |
| 6 | Button | Refresh list of Ports |
| 7 | Button | Accept information |
| 8 | Button | Cancel action |
|  |  |  |
|  |  |  |

Table D‑9 Configuration Interface Design Description

### Object Extractor

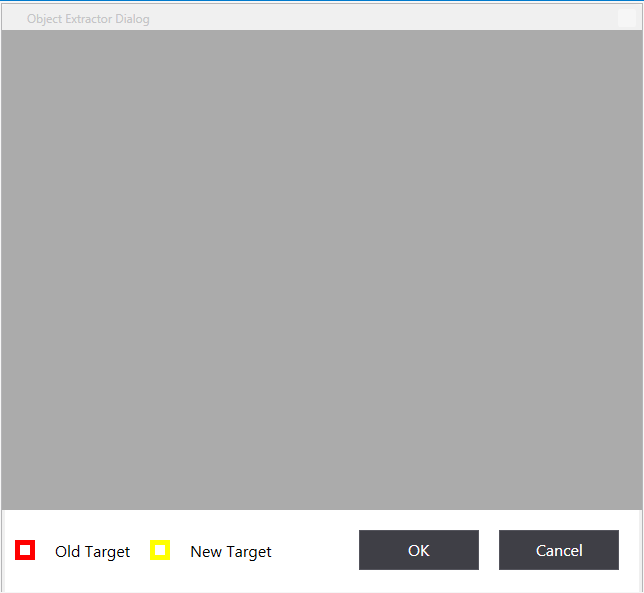


Figure D‑57 Object Extractor Interface Design

|  |  |  |
| --- | --- | --- |
| Item NO. | Component Name | Description |
| 0 | Image | Capture frame for template selection |
| 1 | Button | Accept new template |
| 2 | Button | Cancel action |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Table D‑10 Object Extractor Interface Design Description

# SOFTWARE TEST DOCUMENTATION (STD)

# SOFTWARE USER’S MANUAL (SUM)

# APPENDIX

## Glossary

## References

*All you need to know about AVR fuses* . (2013, May 30). Retrieved from Embedds: http://www.embedds.com/all-you-need-to-know-about-avr-fuses/

Atmel. (n.d.). *ATmega328.* Retrieved from Atmel Web site: http://www.atmel.com/devices/atmega328.aspx?tab=documents

Electrical and Computer Engineering. (n.d.). *C Coding Standard* . Retrieved from Electrical and Computer Engineering Web site: http://users.ece.cmu.edu/~eno/coding/CCodingStandard.html

Frye, J. (2010, June 16). *SSC-32 Manual*. Retrieved from Lynxmotion Web site: http://www.lynxmotion.com/images/html/build136.htm

ITead Studio. (n.d.). *Ultrasonic Ranging Module HC-SR04.* Retrieved from ITead Studio Web site: http://imall.iteadstudio.com/im120628012.html

NXP . (n.d.). *74HC595PW.* Retrieved from NXP Web site: http://www.nxp.com/products/logic/shift\_registers/74HC595PW.html

Oricom Technologies . (2003, January). *Analysis of Multi-Legged Animal + Robot Gaits*. Retrieved from Oricom Technologies Web site: http://www.oricomtech.com/projects/leg-time.htm#Hex2

Tushar. (2013, January 25). *Controlling servo motor with AVR* . Retrieved from Embedds: http://www.embedds.com/controlling-servo-motor-with-avr/

Tushar. (2013, August 4). *Interfacing Ultrasonic Rangefinder with AVR* . Retrieved from Embedds: http://www.embedds.com/interfacing-ultrasonic-rangefinder-with-avr/

1. 1 man-month equals to 22 man-day [↑](#footnote-ref-1)