



國立中山大學電機工程學系

碩士論文

Department of Electrical Engineering

National Sun Yat-sen University

Master Thesis

互動式機器人人偶劇場之編輯平台

An Editorial Platform for Screenplay of Interactive Robotic

Puppet Shows

研究生：郭又齊

Yu-Chi Kuo

指導教授：黃國勝 博士

Dr. Kao-Shing Hwang

中華民國 103 年 2 月

February 2014

國立中山大學研究生學位論文審定書

本校電機工程學系碩士班

研究生 郭又齊 (學號：M993010143) 所提論文

互動式機器人人偶劇場之編輯平台

An Editorial Platform for Screenplay of Interactive Robotic
Puppet Shows

於中華民國 103 年 1 月 28 日經本委員會審查並舉行
口試，符合碩士學位論文標準。

學位考試委員簽章：

召集人 柯登旗 委員 黃辛傳
委員 陳昱仁 委員 黃辛傳

指導教授 黃國燦



學年度：102
學期：1
校院：國立中山大學
系所：電機工程學系
論文名稱(中)：互動式機器人人偶劇場之編輯平台
論文名稱(英)：An Editorial Platform for Screenplay of Interactive Robotic
學位類別：碩士
語文別：Eng
學號：M993010143
提要開放使用：是
頁數：96
研究生(中)姓：郭
研究生(中)名：又齊
研究生(英)姓：Kuo
研究生(英)名：Yu-Chi
指導教授(中)姓名：黃國勝
指導教授(英)姓名：Kao-Shing Hwang
關鍵字(中)：娛樂型機器人
關鍵字(中)：機器人人偶劇場
關鍵字(中)：時間軸
關鍵字(英)：Entertainment robot
關鍵字(英)：Robotic puppet shows
關鍵字(英)：Timeline

中文提要

近年來，在電腦與網路的發展下機器人產業逐漸成長。除學術應用外，娛樂型機器人帶給我們全新的表演方式與娛樂。機器人人偶劇場為其中重要的發展議題。因機器人人偶劇場除整合機器人基本的語音與動作，更具備傳達人類情感的能力。本論文基於以上的考量，開發一套機器人劇場編輯平台，目的為使不同年齡層的使用者，皆能以簡易且快速的方式來操控機器人並設計能讓機器人演出使用者設計的劇本。本平台開發上加入遊戲設計的概念，使劇本內容的編輯上能具備有分歧且多元的特性。最後，本平台透過時間軸驅動控制演戲的流程並藉此解決同步問題，使劇本內容的規劃更具準確性與彈性。從實驗的結果，可看出利用本平台編輯劇本所具備的簡易性與彈性。本論文之成果將以影片呈現，請至 YouTube 網站搜尋 “An Editorial Platform for Screenplay of Interactive Robotic”。

英文提要

In recent years, due to the development of computers and the Internet, the robotics industry has steadily grown. Besides academic applications, entertainment robots can also be programmed to bring us unprecedented performances that many people can enjoy. Robotic puppet shows are an important part of the entertainment robot field. The robot can integrate the basic ability to speak, make poses and human like expressions. Based on the considerations above, this thesis describes an Editorial Platform for Screenplay of Interactive Robotic Puppet Shows (EPFS). EPFS lets users in different age groups control robots and allows them to write the screenplays for drama performances in a simple and fast way. In the platform users have the ability to use game design concepts that allow them to make diverse screenplays and a multitude of different storylines. Finally, the platform controls and records the story-making process by constructing a timeline. In doing so, problems caused by robotic expression and interaction can be solved. In addition, the arrangement of screenplays will be more flexible and accurate. According to the results of the experiment the platform proves to be flexible and easy to use. A video presentation of the thesis has been posted on YouTube, please search the title below “An Editorial Platform for Screenplay of Interactive Robotic”.

致謝

首先，在此由衷地感謝教授黃國勝老師在論文完成的過程中給予許多幫助與詳盡的指導，老師總是能以深入淺出的方式來帶領我們探究學術理論，除專業的知識外，其親切、自由且多元的領導方式讓我印象深刻且獲益良多。其次，感謝陳昱仁老師及惟丞學長在我遭遇研究困難時給予寶貴的建議與幫助。接著我要感謝實驗室成員智中、旻章與宇安不厭其煩地與我討論，讓我能夠快速地掌握機器人人偶劇場之相關研究並加速平台的銜接與開發。其他成員包括豐全、哲勛、仲凱、凡瑋、晁瑋、力維與秀綺以及學弟妹天毓、科男、明翰、嘉舜、超鵬、佐軒、宏軒、炫儀、喬涵與怡家感謝你們與我分享生活中的喜怒哀樂、處理各種事務，讓我的研究生生活多采多姿。此外，特別感謝曾參與機器人人偶劇場表演的學弟妹們，若沒有你們的付出與協助，劇場絕對無法如期完美地演出。接著要感謝女友于萱，在研究過程中總是給予鼓勵、包容並體諒我。最後，我要感謝父母一路以來的栽培與支持，並給我許多空間去自行發揮。在我最需要幫助的時候提供一個溫暖的避風港讓我保有持續向前的力量。在這兩年多的研究生涯中充滿各種挑戰，其中包含機器人相關競賽、研討會發表與劇場展示，讓我除了明白自身的不足與渺小之外，亦深切地體悟到「學海無涯，為勤是岸」的道理。其中，在競賽與展示過程中藉著夥伴們之間互相幫助，最終排除萬難達到目標的那份喜悅與成就感至今仍然讓我深深地感動，這些美好的經驗將成為我人生中不可或缺的寶藏。最後，再次感謝所有曾經提供協助的貴人們，謝謝你們！

又齊 謹誌於

中山大學電機工程研究所
智慧型機器人及訊息系統實驗室

摘要

近年來，在電腦與網路的發展下機器人產業逐漸成長。除學術應用外，娛樂型機器人帶給我們全新的表演方式與娛樂。機器人人偶劇場為其中重要的發展議題。因機器人人偶劇場除整合機器人基本的語音與動作，更具備傳達人類情感的能力。本論文基於以上的考量，開發一套機器人劇場編輯平台，目的為使不同年齡層的使用者，皆能以簡易且快速的方式來操控機器人並設計能讓機器人演出使用者設計的劇本。本平台開發上加入遊戲設計的概念，使劇本內容的編輯上能具備有分歧且多元的特性。最後，本平台透過時間軸驅動控制演戲的流程並藉此解決同步問題，使劇本內容的規劃更具準確性與彈性。從實驗的結果，可看出利用本平台編輯劇本所具備的簡易性與彈性。本論文之成果將以影片呈現，請至YouTube網站搜尋“An Editorial Platform for Screenplay of Interactive Robotic”。

關鍵詞：娛樂型機器人、機器人人偶劇場、時間軸

ABSTRACT

In recent years, due to the development of computers and the Internet, the robotics industry has steadily grown. Besides academic applications, entertainment robots can also be programmed to bring us unprecedented performances that many people can enjoy. Robotic puppet shows are an important part of the entertainment robot field. The robot can integrate the basic ability to speak, make poses and human like expressions. Based on the considerations above, this thesis describes an Editorial Platform for Screenplay of Interactive Robotic Puppet Shows (EPFS). EPFS lets users in different age groups control robots and allows them to write the screenplays for drama performances in a simple and fast way. In the platform users have the ability to use game design concepts that allow them to make diverse screenplays and a multitude of different storylines. Finally, the platform controls and records the story-making process by constructing a timeline. In doing so, problems caused by robotic expression and interaction can be solved. In addition, the arrangement of screenplays will be more flexible and accurate. According to the results of the experiment the platform proves to be flexible and easy to use. A video presentation of the thesis has been posted on YouTube, please search the title below “An Editorial Platform for Screenplay of Interactive Robotic”.

Keywords : Entertainment robot, Robotic puppet shows, Timeline

TABLE OF CONTENTS

摘要	v
ABSTRACT	vi
LIST OF FIGURES	ix
LIST OF TABLES	xi
I. INTRODUCTION.....	1
1.1 Motivation	1
1.2 Objective.....	3
1.3 Organization of thesis	4
II. BACKGROUND	5
2.1 System Environment and Development Tools	5
2.1.1 MVC Architecture	5
2.1.2 System Environment	6
2.1.3 The User Interface Framework.....	10
2.1.4 Networking between Platform and Robots.....	10
2.2 Motion Capture.....	12
2.3 Related Works.....	14
III. SYSTEM ANALYSIS AND DESIGN	15
3.1 System Introduction.....	15
3.2 System Analysis.....	19
3.2.1 5W1H Method	19
3.2.2 Functional Requirements	22
3.2.3 Non-functional Requirements	23
3.2.4 Use Cases.....	23
3.3 Motion Generation.....	31
3.3.1 Preprogramed Robotic Motion	31
3.3.2 Kinect-based Motion Uploading	33
3.4 Screenplay Editing.....	41
3.4.1 Scene Tree	41
3.4.2 Time Stamp.....	42
3.4.3 Screenplay Format	44
3.5 Screenplay Performance	47
3.5.1 Interactive Screenplay Authoring Platform (ISAP).....	47
3.5.2 Mobile Interpreter (MI)	56
3.6 Hardware Architecture.....	58
3.6.1 DARwIn-OP	58
3.6.2 Bioloid	59

3.6.3 NAO	60
3.6.4 Smart Device	61
3.7 Software Architecture	63
3.7.1 DARwIn-OP API	63
3.7.2 NAO API	68
3.7.3 Tablet Computer	69
IV. IMPLEMENTATION	71
4.1 Development Tools	71
4.2 Implementation of ISAP	73
4.3 User Interface of ISAP	75
4.4 Android Application for Robotic Puppet Show	87
4.5 Improvement	88
V. CONCLUSION AND FUTURE WORK	92
5.1 Conclusion	92
5.2 Future Work	93
REFERENCE	95

LIST OF FIGURES

Figure 2-1 The basic MVC relationship	6
Figure 2-2 Market share of active sites.....	7
Figure 3-1 Internet popularity growth chart	16
Figure 3-2 The concept of EPFS.....	17
Figure 3-3 Screenplay performance process.....	18
Figure 3-4 EPFS map.....	18
Figure 3-5 Use case diagram of EPFS	30
Figure 3-6 RoboPlus interface	31
Figure 3-7 Choregraphe interface	32
Figure 3-8 NITE skeleton tracing algorithm	33
Figure 3-9 Coordinate relationship diagram.....	35
Figure 3-10 PRY convention	37
Figure 3-11 Key-pose selection flow chart.....	40
Figure 3-12 Posture within Kinect-based motion file.....	41
Figure 3-13 Structure of scene tree.....	42
Figure 3-14 Timeline tracks within a scene	44
Figure 3-15 JSON forms.....	45
Figure 3-16 Screenplay formatted using JSON	46
Figure 3-17 The online JSON parser	46
Figure 3-18 EPFS architecture.....	47
Figure 3-19 Website based on MVC.....	48
Figure 3-20 ISAP architecture	49
Figure 3-21 Polling and WebSocket latency comparison	50
Figure 3-22 The process of performing a screenplay	51
Figure 3-23 Command execution by Node.js via stream sockets.....	52
Figure 3-24 Multiple child nodes cause branching.....	53
Figure 3-25 Node.js process	53
Figure 3-26 The format of the type I packet.....	54
Figure 3-27 The format of the type II packet.....	55
Figure 3-28 The format of the type III packet	56
Figure 3-29 MI architecture	57
Figure 3-30 DARwIn-OP hardware.....	58
Figure 3-31 Front and rear view of Bioloid.....	59
Figure 3-32 Bioloid applications	59

Figure 3-33 Bioloid equipped with USR-WIFI232-2 module.....	60
Figure 3-34 Hardware framework of NAO	61
Figure 3-35 Appearance of ASUS MEMO Pad FHD 10.....	62
Figure 3-36 The cycle of a thread.....	64
Figure 3-37 Hue scale.....	66
Figure 3-38 Image via camera	67
Figure 3-39 Android architecture.....	69
Figure 4-1 The process of copying the Python code	72
Figure 4-2 Pasting the Python code into console.....	72
Figure 4-3 Run the related commands via packet control	72
Figure 4-4 Android SDK installment.....	73
Figure 4-5 Entity relationship diagram for EPFS database	74
Figure 4-6 Website flow chart.....	75
Figure 4-7 Log into website.....	75
Figure 4-8 Home page	76
Figure 4-9 User tutorial	76
Figure 4-10 Screenplay creation.....	77
Figure 4-11 List of screenplays.....	78
Figure 4-12 Adding a new scene	79
Figure 4-13 Scene box functions	79
Figure 4-14 Scene-editing page	80
Figure 4-15 Adding a new motion into a track.....	81
Figure 4-16 Speech function.....	81
Figure 4-17 Linking tool of scene	82
Figure 4-18 Branching	83
Figure 4-19 Motion management page.....	83
Figure 4-20 Motion adding page for DARwIn-OP.....	84
Figure 4-21 Motion adding page for Bioloid.....	84
Figure 4-22 Motion adding page for new type robot.....	85
Figure 4-23 Uploading a Kinect-based motion file	86
Figure 4-24 Running Node.js	86
Figure 4-25 Setting IP addresses and port numbers.....	87
Figure 4-26 Screenplay list on tablet	88

LIST OF TABLES

Table 2-1 The usage rates of different web server applications.....	8
Table 2-2 Comparison with databases	9
Table 2-3 Comparison with wireless technologies	11
Table 2-4 Comparison of motion sensing devices	12
Table 2-5 Comparison between OpenNI and Kinect for Windows SDK	13
Table 3-1 5W1H analysis.....	21
Table 3-2 Use case of create a new screenplay.....	23
Table 3-3 Use case of delete an existing screenplay.....	23
Table 3-4 Use case of remand the annotations of existing screenplay	24
Table 3-5 Use case of specify a screenplay to edit	24
Table 3-6 Use case of create a new scene.....	24
Table 3-7 Use case of delete an existing scene.....	25
Table 3-8 Use case of make a link between different scenes.....	25
Table 3-9 Use case of delete a link between different scenes.....	26
Table 3-10 Use case of specify a scene to edit	26
Table 3-11 Use case of arrange the motion a robotic puppet.....	26
Table 3-12 Use case of arrange the line of a robotic puppet.....	27
Table 3-13 Use case of arrange the sound effect	27
Table 3-14 Use case of arrange the background music	28
Table 3-15 Use case of expand the motion library	28
Table 3-16 Use case of add new robot to existing system	28
Table 3-17 Use case of perform the robotic puppet show	29
Table 3-18 Use case of pick different paths as the story progresses.....	29
Table 3-19 The ASUS MEMO Pad FHD 10 product specification	62
Table 3-20 Result of color detection.....	68
Table 4-1 Comparison between proposed platforms I, II and EPFS	91

I. INTRODUCTION

1.1 Motivation

In today's world Robots are replacing humans in many jobs that are too trivial or dangerous for people to do. The introduction of robots into our society has begun to make many people's lives more convenient and safer than ever. The use of robotics can be applied to academic and industrial applications as well as many other fields. Service and entertainment robots have gradually made their entrance into our daily lives. This paper discusses the development an Editorial Platform for Screenplay of Interactive Robotic Puppet Shows (EPFS) which applies to multiple-type robots that are used in the entertainment field. The graphic and intuitional user interfaces are easy enough for a child to operate. This paper discusses a platform that has been developed to control multiple-types of robots in a simple way. Because the platform can control more than one robot at the same time it is a suitable technology to be used in robotic puppet shows which are like real dramas.

Dramas and games have many similarities; they are both forms of art that can also be enjoyed as recreational activities. The biggest difference between drama and games is that a drama is passive, and game is interactive. The content of drama is directly presented to the audience, in the form of television or movies. In dramas the audience only can watch as the story unfolds. In contrast, games allow users to take

part in the story, which as a result could make the ending of the story different. In short, playing games is just about making continuous choices. For example, in Massively Multiplayer Online Role-playing Game (MMORPG) each player's role is the same when they first start playing. As each individual player makes their own choices these players cause each character in the story to have a distinctly different fate. Choices can be split into two types, meaningless and meaningful. Meaningless choices include the ability to change the gender and appearance of a character. These details may have little or no effect on the outcome of the story itself. On the other hand, meaningful choices have critical effects on the direction of the story. In addition, during story development there are scripted events and triggered events [1]. Scripted events are previously arranged and they cannot be changed by the choices that characters make. For example: weather conditions such as thunder, lightning or rain cannot be changed by any individual character. However, triggered events only occur if the character satisfies certain requirements, which will affect the next part of the story. For example, Mark Zuckerberg, the founder of Facebook, is very talented at programming. With an adventurous personality, he posted schoolmate's photos on a website he made without permission. Little did he know that doing this would eventually lead him to establish the popular social networking website called Facebook. If Mark had been reserved and serious, he would never have posted those

photos and Facebook would not have been created. Thus, people make choices depending on their personality. Today's robotic puppet shows usually use event-driven story arrangements [2][3] that are too simple to be used in situations where sophisticated functions are needed. In this thesis, the time stamps within timelines are implemented in order to arrange expressions in motion and speech and the interaction between different robotic puppets and music.

1.2 Objective

The concept of EPFS is to let users in different age groups edit their own personal screenplays just by accessing the website. By editing the screenplay users can change the robots posture and speech to make their expressions more humanlike. The joining of sound effects and music can also make the show more dramatic. After finishing a screenplay, the development and ending of a drama are fixed. A new feature called "branching" has been added to the platform which allows the user to pick different paths as the story progresses. This makes the show more intriguing and multi-dimensional. In addition, problems of robotic expression and interaction are solved by using timeline to arrange the process of performance.

1.3 Organization of thesis

This thesis is divided into five chapters. The first chapter explains the motivation and objective of the thesis. This platform not only provides users with powerful and easy to use features, it also has joined concepts from gaming with robotic puppet shows. The second chapter describes the platform environment, development tools and related works. The third chapter is system analysis and design, this chapter also defines the motion generation, screenplay editing, script performance and introduction to the hardware and software architecture used in the platform. Chapter 4 presents the implementation of the platform and compares previous works. Chapter 5 summarizes this thesis and makes some recommendations for future work.

II. BACKGROUND

In this chapter, we introduce the environment of system and tools that could be utilized to develop a web based platform. Furthermore, motion capture technology is introduced as well as other proposed platforms for the robotic puppet show.

2.1 System Environment and Development Tools

2.1.1 MVC Architecture

In recent web development, a single web page is usually written with different languages such as HTML, CSS, and PHP. Because of this, the source code will be difficult to maintain or modify in the future.

MVC is a design pattern for software engineering as show in Figure 2-1 [4]. It divides an application into three parts: Model, View and Controller. MVC was designed by Trygve Reenskaug in 1979. The model-view-controller pattern is mainly used to simplify code modification, extensibility and reutilization. The model is applied to deal with business logic and interface data exchange. Of the three parts within the pattern, only the model can access the database directly. The view can develop visual outputs for users, which is used for data display, such as a webpage.

The controller is responsible for collecting the input data from the view in order to translate it into commands for updating the state of models or changing the corresponding view's appearance.

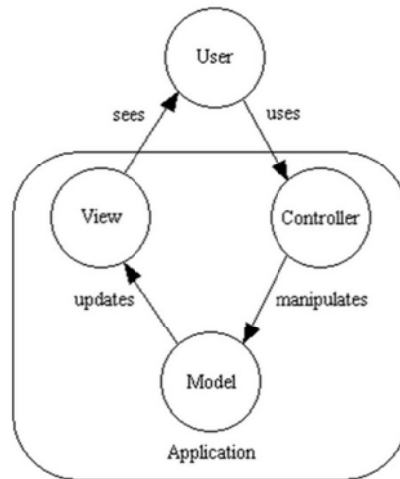


Figure 2-1 The basic MVC relationship

2.1.2 System Environment

LAMP is a software bundle that is used for building web servers. The name stands for Linux, Apache, MySQL and PHP. LAMP has become one of the popular web platforms around the world because it is a free, stable and open source software package.

(1) Ubuntu

Ubuntu is an operating system based on Linux that was designed to be user friendly. It can be divided into two versions: the server and desktop. The main difference between them is that the desktop is built with a GNOME (GNU Network Object Model Environment) that helps the user to operate the system by using easily

interpretable graphics. On the contrary, the server only provides users with text operations. Below are the main features of Ubuntu:

- Stable and fast
- Easy installation and user-friendly interface
- Plenty of free software
- High degree of freedom

(2) Apache

Apache (Apache HTTP Server) is a web server application that is used to provide computer browser service. Besides being free of charge Apache also has high security, stability and extensibility. In addition, it can be run in different operating systems such as UNIX, Linux and Windows. Figure 2-2 [5] shows a survey of active sites by Netcraft, which indicates that Apache has been the most popular web server application. The usage rate of Apache by web server developers was 53.96% in December 2013 and 54.50% in January 2014 in Table 2-1 [5].

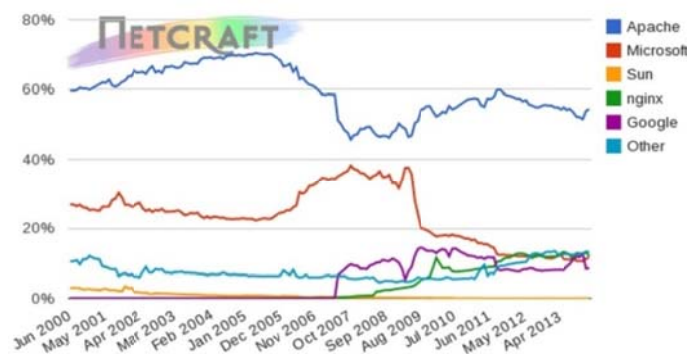


Figure 2-2 Market share of active sites

Table 2-1 The usage rates of different web server applications

Developer	December 2013	Percent	January 2014	Percent	Change
Apache	99,367,323	53.96%	98,129,017	54.50%	0.54
Microsoft	24,534,598	13.32%	21,548,550	11.97%	-1.36
nginx	20,731,750	11.26%	20,901,626	11.61%	0.35
Google	15,508,986	8.42%	15,386,518	8.54%	0.12

(3) MySQL

In order to build dynamic websites and applications, a database is used to store a variety of important data. For example, since the user has entered the website, the user account, password should be stored in the database. The screenplay details that have been changed or modified on the webpage should be saved as well.

MySQL is a relational database management system that provides a cost efficient and highly reliable solution for database construction and management. In addition, table 2-2 [6] shows that MySQL has the highest operating system support. It also has less data restriction and a GPL (General Public License) that gives the developer more freedom to manipulate the software.

Table 2-2 Comparison with databases

Function Name	OS support	Max DB size	Maintainer	Latest release date	Software license
MS Access	Low	2GB	Microsoft	2012-10-02	Proprietary
PostgreSQL	Medium	Unlimited	PostgreSQL Global Development Group	2013-12-05	PostgreSQL Licence
Oracle	Medium	Unlimited	Oracle Corporation	2013-06-25	Proprietary
MySQL	High	Unlimited	Oracle Corporation	2013-07-30	GPL or Proprietary
SQLite	High	128 TB	D. Richard Hipp	2013-09-03	Public domain
DB2	Medium	Unlimited	IBM	2013-04-23	Proprietary

(4) PHP

PHP (Hypertext Preprocessor) is an open source, cross-platform scripting language that is widely used in web development. The major goal is to allow developer to create dynamic webpages quickly. PHP can be run in various operating systems, web servers and databases. The PHP interpreter has been installed on the web server in order to generate the corresponding web page by using the PHP code (even when it is embedded in an HTML document). In addition, PHP provides inherent functions used to communicate with the MySQL database which enhances them. This combination has become popular for building dynamic websites.

2.1.3 The User Interface Framework

JavaScript is a scripting language that can be divided into two parts: the client-side and server-side. For normal usage, JavaScript is usually embedded with an HTML document. It is recognized as client scripts that are directly interpreted by the web browser rather than sending data back to the web server. It is mainly used for user interaction such as event reaction, data validation and HTML elements reading and writing.

JQuery is a JavaScript library that provides a powerful selector for picking all DOM elements or the partial parts that are based on designated ID or CSS in order to bind various operations together such as event triggers, animations, effects, AJAX, and extensibility through plug-ins. In this platform the Kendo UI, a jQuery based framework includes many UI widgets and data visuals in order to make the web interface more attractive and user-friendly.

2.1.4 Networking between Platform and Robots

A wireless network was used to create the connection between the platform and the robots because this kind of system gives the robot the largest range of maneuverability. Table 2-3 shows the differences among the three existing common wireless networks that may be suitable for use in robot control.

Table 2-3 Comparison with wireless technologies

Function \ Wireless tech.	Wi-Fi	Bluetooth	ZigBee
Standard	802.11.b	802.15.1	802.15.4
Range	100 meters	10-100+ meters	10-100+ meters
Data Rate	11Mbps	1Mbps	250kbps(2.4GHz) 40kbps(915MHz) 20kbps(868MHz)
Frequency	2.4 GHz	2.4 GHz	2.4 GHz 868/915 MHz
Battery Life (hours)	AC power	1-7	100+
Number of Network Nodes	32	8	65535
Major Advantages	High speed Flexibility	Low Cost Speech function	Low Cost Low Power Consumption
Typical Applications	Wireless LAN connectivity	Replaces wire connectivity	Remote control

Although the physical assessment range (100m) for wireless is very close to the range available for the different kinds of wireless technologies in the study, Wi-Fi has shown to be the fastest of these technologies. Despite the fact that Wi-Fi has the highest power consumption of all of the wireless technologies researched in this study, Wi-Fi technology has still matured and been widely used as a way for laptops and smart devices to access the Internet. The platform that was used during the research project was connected to an Android application that puts the performance robots into play. Most important of all, Wi-Fi can let our smart devices such as smartphones or tablet computers easily communicate with the platform and robots.

2.2 Motion Capture

In the robotic puppet show, the developer looks forward to making the robotic motion more vivid and human-like. But editing robotic motion was usually done by adjusting the parameters of the robot's actuators.

Using this process is repetitive and inefficient. In [7], the motion sensing devices based on optics such as Microsoft Kinect, and Asus Xtion Pro Live. That provides users with an accurate, cost-effective and state of the art functions to capture human motions. By using sensing devices the motions generated by imitation can be used to update the online motion library for the robotic puppet show.

Table 2-4 Comparison of motion sensing devices

Devices	Kinect	Xtion Pro Live
Functions		
Depth Information	Yes	Yes
RGB	Yes	Yes
Frame Rate	30 frames per second	30 frames per second
Audio Input	A four-microphone array	Microphone
API	OpenNI / Microsoft SDK	OpenNI
Tilt Motor	Yes	No
Volume	Larger	Smaller
Price	Low	High

Table 2-4 shows that Kinect and Xtion Pro Live are designed to acquire RGB, depth and audio streams. Both of them have RGB camera, a pair of infrared devices, and microphones. The RGB sensor is used to capture pictures based on the color

channel data. The pair of IR devices including the emitter and the depth sensor work together to emit infrared light beams and sense the reflected infrared which will be translated into depth images by PrimeSense's Light Coding technology. The microphone allows users to trace the source of sounds and develop applications for voice configuration. Furthermore, Kinect also has an additional tilt motor used for vertical adjustment of the lens

Based on the information collected from the motion sensing devices, the APIs provide the developer with related functions such as skeleton and face tracking [8]. OpenNI and Kinect for Windows SDK are two of the most used APIs. The major difference is that the OpenNI can support Kinect as well as Xtion Live Pro, On the contrary, Kinect for Windows SDK can only be applied to Kinect. Table 2-5 shows the function comparison between these APIs.

Table 2-5 Comparison between OpenNI and Kinect for Windows SDK

Function \ API	OpenNI	Kinect for Windows SDK 1.0
Hand Recognition	Yes	No
Joint Position	Yes	Yes
Joint Orientation	Yes	No
Record/Replay	Yes	No
User Event Notification	Yes	No
OS Support	Windows/Linux/Mac	Windows 7

2.3 Related Works

In [2][3], J.J. Siao and M.C. Wu have proposed a web-based platform designed to perform the robotic puppet show by using the DARwIn-OP and NAO robots.

The online website also allows users to edit personal screenplays for performance purposes. Each screenplay is composed of lots sequential events, which can be updated by adding one designated robot's default motion or filling in text. Moreover, those events can be added, deleted and rearranged. Any two different screenplays can be merged in order to generate new one.

The authoring tool and platform for robotic puppet shows was developed by C.A. Chen [9]. It was built as a cloud service to allow users to compose their own unique screenplays through the Internet. While editing the screenplay, users are allowed to edit each puppets performance including posture adjustment, line adding and emoticon settings within each time stamp. Moreover, the smart phones are equipped with the robotic puppets (Bioloid) as controllers. So that the puppets can receive the instructions generated by a drama object and implement the corresponding functions. There are two different kinds of play modes: single mode and synchronous mode. Single mode only allows the director to control one puppet at a time. With the use of synchronous mode, multiple puppets can perform at the same time.

III. SYSTEM ANALYSIS AND DESIGN

In this chapter, the system introduction is illustrated, and then the 5W1H analysis method will be applied to distinguish the platform's capabilities including the functional and non-functional requirements. The screenplay performance process including the basic motion generation, scene and screenplay editing as well as the script performance will be described in the following sections. As a result, the hardware and software architectures will be depicted in order to represent how to work with the platform.

3.1 System Introduction

In general, a controller is designed to control a single type of robot. Therefore, controlling different types of robots at the same time would be a complicated and high-cost task. This thesis develops a system to solve the previously mentioned problem and prove it by using a robotic puppet show. The core design concept is to allow users of different ages to easily edit their own screenplays and to make different kinds of performance robots act in the edited screenplays.

According to the survey by the Institute for Information Industry in Figure 3-1 [10], the domestic population of Internet users has reached 11 million and seventy thousand in the fourth quarter of 2012. Web service has become an essential part of

human life. Thus, developing a system to become a web application that uses a web browser as a client, allows this application to be easily and widely used by the public. By using the web based feature, restrictions such as distance and time can be broken and less storage space is needed on the client.

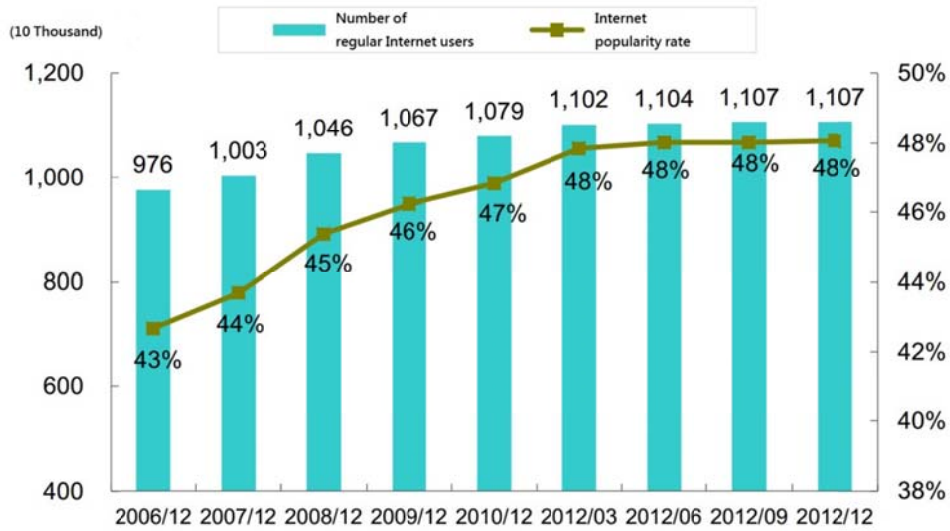


Figure 3-1 Internet popularity growth chart

The wireless technology, Wi-Fi, not only has a high transport speed but also has commonly used in daily life. In addition, the robotic puppets have a built in Wi-Fi function or can be equipped with a Wi-Fi module. Therefore, Wi-Fi is chosen as the main communication type in this system to allow users to have the ability to access the website and control robotic puppets. The concept of system is described in Figure 3-2.

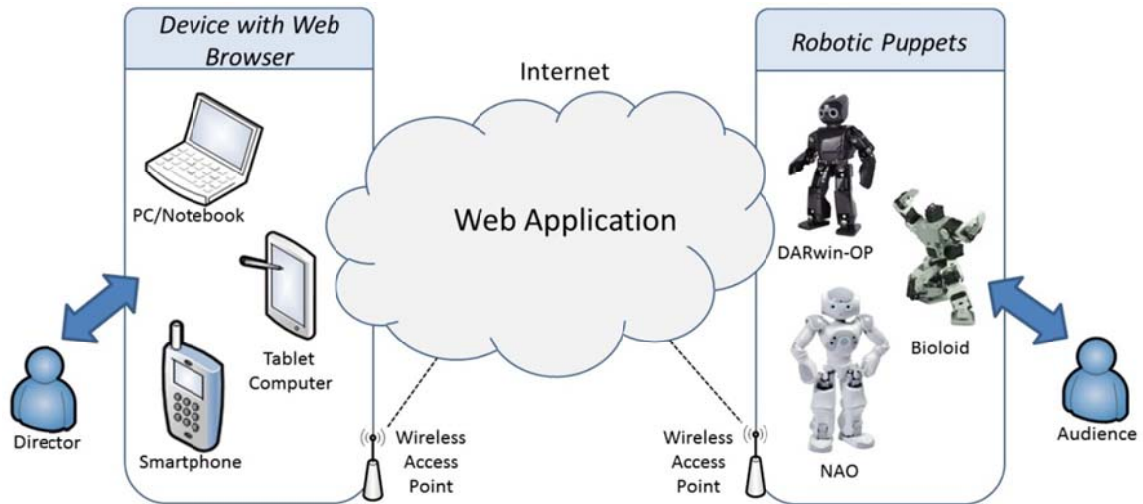


Figure 3-2 The concept of EPFS

The process of screenplay performance can be drawn as Figure 3-3. Three ways to generate motions for controlling robots: actuator parameter adjustment, preprogrammed robotic motions and Kinect-based motions.

Actuator parameter adjustment [9] is commonly used to control the robots joints. In this thesis, preprogrammed robotic motions and Kinect-based motions are available in robotic puppet shows. The preprogrammed robotic motions are developed and stored within the robotic puppets for future usage. In addition, the platform also allows users to generate new motions by using Kinect sensing devices.

Next, the two previously mentioned motions can be joined into the motion library and arranged within the timeline of each scene. At that time, the scenes can be organized and used as the content of the screenplay. As a result, the screenplays can be performed and shared by any user who has signed up for the platform. The whole map of EPFS can be illustrated as Figure 3-4. This platform contain two different

parts: the Interactive Screenplay Authoring Platform (ISAP) and the Mobile Interpreter (MI). The ISAP provides users with six different function webpages: home, tutorial, screenplay editing, motion management, Kinect-based motion uploading, screenplay performance, and contact. On the other hand, MI only provides the screenplay performance function.



Figure 3-3 Screenplay performance process

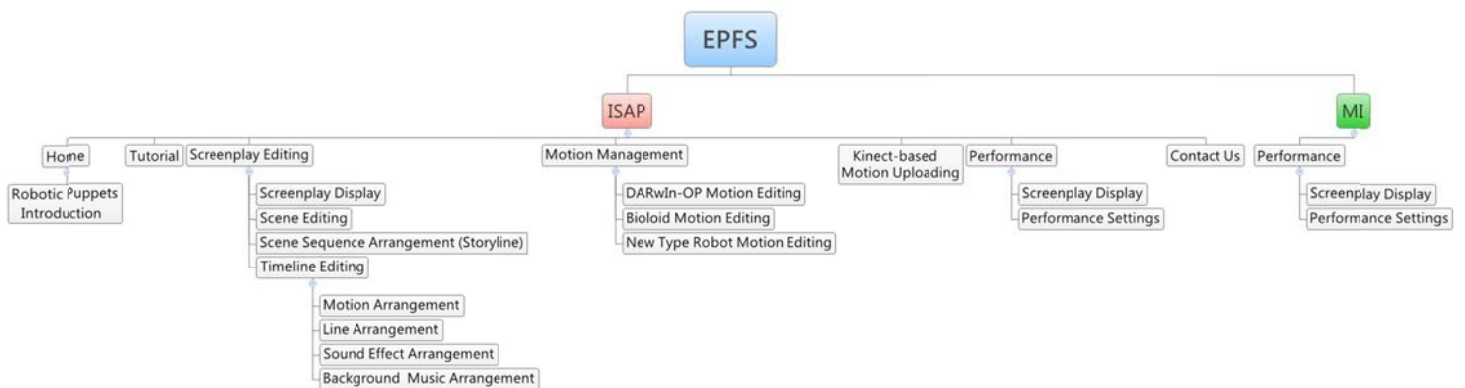


Figure 3-4 EPFS map

3.2 System Analysis

3.2.1 5W1H Method

The following content use 5W1H analysis to analyze the platform, and the summary of 5W1H was shown in Table 3-1.

The first W is why—Why design the platform? Since different robots have a variety of control ways, the platform is designed to provide even basic user has the ability to control robotic puppets in a fast and simple way. The second reason is for education. The graphical user interface (GUI) of platform can make user easily use even the children [11]. As a teacher, who can edit and play a story for lesson. In the other hand, children can learn to how to be a good story-teller or just use different color cards to make an interaction with robots. In addition, robotic puppet show also brings people the brand new entertainment.

The second W is what—What functions of the platform are designed for user? The platform is mainly designed to make user have the ability to edit and play the screenplay for robotic puppet show in a fast and simple way via GUI. Secondly, the platform allow user to control any kinds of performance robot. The timeline is be used within each scene in order to solve the problem of synchronization, which allow user to precisely arrange the performance elements including motion, speech, and sound. Unlike normal drama which has the sequential way to perform, dynamic interaction

make the story diversify according to the result of color detection. Finally, the platform uses Kinect to capture human motion in order to update the motion library of performance robot.

The third W is where—Where can the platform be use? For considering the performance requirement, the platform is designed with portability. With Internet access, the robot puppets show can perform in any place in order to let more people enjoy it.

The forth W is when—When can the platform be use? Any time if user want to perform the show, all he/she needs to do is to make sure the web device have the ability to access Internet, and the performance robots have get ready.

The fifth W is who—Who is to use the platform? People in different age can easily get familiar to operate the platform. Even the children who doesn't have a fully ability to completely edit the whole screenplay, they still have fun with robots in interaction feature.

The last H represents how—How to use the platform ? User needs use any device with browser to access the Internet for requesting the detail of screenplay. The web device and performance robots need to be connected in a LAN in order to communicate with each other by WI-FI technology.

Table 3-1 5W1H analysis

<u>Why</u> design the platform?	<ol style="list-style-type: none"> 1. Control different type robots in a simple way. 2. Education application and brand new entertainment.
<u>What</u> functions of the platform are designed for user?	<ol style="list-style-type: none"> 3. Edit a screenplay for robotic puppet show. 4. Control any types of performance robots. 5. Dynamic interaction 6. Add new robot motion for drama via Kinect.
<u>Where</u> can the platform be use?	Anywhere if the user has the web device with Internet.
<u>When</u> can the platform be use?	Any time if the user has the web device with Internet.
<u>Who</u> is to use the platform?	For all ages, especial for children.
<u>How</u> to use the platform?	<ol style="list-style-type: none"> 1. Any device that has the ability to access Internet via bowser. 2. Through WI-FI technology.

3.2.2 Functional Requirements

- (1) Create a new screenplay.
- (2) Remand the annotations of existing screenplay.
- (3) Delete an existing screenplay.
- (4) Specify a screenplay to edit.
- (5) Create a new scene in a screenplay.
- (6) Delete an existing scene.
- (7) Make a link between different scenes.
- (8) Edit a scene's content.
- (9) Delete a link between different scenes.
- (10) Update the motion library by Kinect.
- (11) Add a new type robot.
- (12) Perform a show.

3.2.3 Non-functional Requirements

- (1) The Graphic User Interface make the platform be easily realized and used.
- (2) Dragging and clicking the elements of website allow user to finish the complicated and redundant task in an intuitive and efficient way.
- (3) Join the audio element such as sound effect and background music to enhance the dramatic tension.

3.2.4 Use Cases

Table 3-2 Use case of create a new screenplay

Title	Create a new screenplay.
Description	Create a new screenplay with annotations.
Actors	Directors.
Pre-condition	Review the screenplay list page.
Post-condition	The new screenplay box appears in the display list.
Scenario	<ol style="list-style-type: none">1. Log in the website.2. Create a new screenplay.3. Finish the annotations of screenplay including a unique name, cover, description, and each kinds of robot number.4. The box of the new screenplay shows in the display list.

Table 3-3 Use case of delete an existing screenplay

Title	Delete an existing screenplay.
Description	Delete an existing screenplay that is selected.
Actors	Directors.
Pre-condition	Select the screenplay box which is displayed in the list.
Post-condition	The screenplay box disappears in the display list.
Scenario	<ol style="list-style-type: none">1. Log in the website.2. Select the screenplay box in the display list.

	<ol style="list-style-type: none"> 3. Delete it in the display list. 4. The deleted screenplay box vanishes in the display list.
--	--

Table 3-4 Use case of remand the annotations of existing screenplay

Title	Remand the annotations of existing screenplay.
Description	Select an existing screenplay to remand its annotations.
Actors	Directors.
Pre-condition	Select the screenplay box which is displayed in the list.
Post-condition	The annotations of screenplay have been updated.
Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Select the screenplay box in the display list. 3. Remand the screenplay annotations including the name, cover, or description. 4. The content of annotations belonging to the screenplay has been updated.

Table 3-5 Use case of specify a screenplay to edit

Title	Specify a screenplay to edit.
Description	Select a screenplay to edit its scenes.
Actors	Directors.
Pre-condition	Select the screenplay box which is displayed in the list.
Post-condition	Enter the screenplay editing interface.
Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Select the screenplay box in the display list. 3. Enter the screenplay editing interface.

Table 3-6 Use case of create a new scene

Title	Create a new scene.
Description	Create a new scene with annotations in a screenplay.
Actors	Directors.
Pre-condition	The screenplay has been specified.
Post-condition	The new scene box appears within the interface.

Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Select the screenplay box in the list. 3. Enter the screenplay editing interface. 4. Create a scene with its annotations including a name, last time (second), description, and driver type. 5. The box of the new scene shows in the interface.
----------	--

Table 3-7 Use case of delete an existing scene

Title	Delete an existing scene.
Description	Delete an existing scene that is selected.
Actors	Directors.
Pre-condition	Select the scene box which is displayed in the interface.
Post-condition	The scene box disappears in the display list.
Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Select the screenplay box in the list. 1. Enter the screenplay editing interface. 2. Delete the scene as well as all of arrows with it. 3. The deleted scene box vanishes in the interface.

Table 3-8 Use case of make a link between different scenes

Title	Make a link between different scenes.
Description	Determine the perform sequence and branching.
Actors	Directors.
Pre-condition	Pick two different scene boxes which are displayed in the interface.
Post-condition	The arrow of the two scenes appears.
Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Select the screenplay box in the list. 3. Enter the screenplay editing interface. 4. Use link tool to connect two different scenes. 5. The link between the two scenes has been established.

Table 3-9 Use case of delete a link between different scenes

Title	Delete a link between different scenes.
Description	Cancel the perform sequence and branching.
Actors	Directors.
Pre-condition	Pick two different scene boxes which are linked in the interface.
Post-condition	The arrow of the two scenes vanishes.
Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Select the screenplay box in the list. 3. Enter the screenplay editing interface. 4. Double click the arrow to delete it. 5. The link between the two scenes has been canceled.

Table 3-10 Use case of specify a scene to edit

Title	Specify a scene to edit.
Description	Select a scene to edit its story.
Actors	Directors.
Pre-condition	Select the scene box which is displayed within the screenplay.
Post-condition	Enter the scene editing interface.
Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Select the screenplay box in the display list. 3. Enter the scene editing interface.

Table 3-11 Use case of arrange the motion a robotic puppet

Title	Arrange the motion of a robotic puppet.
Description	Add/delete the robot motion on the timeline within a scene.
Actors	Directors.
Pre-condition	The scene has been specified.
Post-condition	Motion box appears/disappears on the timeline.
Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Select the screenplay box in the display list. 3. Enter the scene editing interface. 4. Pick up the motion file and drag it into a track when adding. 5. Double clicking the motion box in a track to delete it.

	6. Motion box appears/disappears with its name on the timeline that represents the starting and ending time.
--	--

Table 3-12 Use case of arrange the line of a robotic puppet

Title	Arrange the line of a robotic puppet.
Description	Add/delete the robot line on the timeline within a scene.
Actors	Directors.
Pre-condition	The scene has been specified.
Post-condition	Line box appears/disappears on the timeline.
Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Select the screenplay box in the display list. 3. Enter the scene editing interface. 4. Click a track to fill in the line. 5. Clear the content of line in the track in order to delete it. 6. Line box appears/disappears with its content on the timeline that represents the starting time.

Table 3-13 Use case of arrange the sound effect

Title	Arrange the sound effect.
Description	Add/delete the sound effect on the timeline within a scene.
Actors	Directors.
Pre-condition	The scene has been specified.
Post-condition	Sound box appears/disappears on the timeline.
Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Select the screenplay box in the display list. 3. Enter the scene editing interface. 4. Pick up the sound file and drag it into a track when adding. 5. Double clicking the sound box in a track to delete it. 6. Sound box appears/disappears with its name on the timeline that represents the starting and ending time.

Table 3-14 Use case of arrange the background music

Title	Arrange the background music.
Description	Add/delete the background music on the timeline within a scene.
Actors	Directors.
Pre-condition	The scene has been specified.
Post-condition	Music box appears/disappears on the timeline.
Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Select the screenplay box in the display list. 3. Enter the scene editing interface. 4. Pick up the music file and drag it into a track when adding. 5. Double clicking the music box in a track to delete it. 6. Music box appears/disappears with its name on the timeline that represents the starting and ending time.

Table 3-15 Use case of expand the motion library

Title	Expand the motion library.
Description	Update the imitation file generated by Kinect to expand the motion library.
Actors	Directors.
Pre-condition	Use Kinect to capture user's motion and record it to a file.
Post-condition	The motion library expands a new motion file.
Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Select the imitation file generated by Kinect and upload it to sever. 3. The new motion appears in the library.

Table 3-16 Use case of add new robot to existing system

Title	Add new robot to existing system.
Description	Control not only default robot but also any kinds.
Actors	Directors.
Pre-condition	The new robot can acts as the format of motions and lines via TCP/IP socket.
Post-condition	The new robot performs with other default robots in show.

Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Add a new robot motion setting including a name, cover, description, command format, last time. 3. The motion file that belongs to the new robot appears in the motion library. 4. Drag the motion file into a track in order to act.
----------	--

Table 3-17 Use case of perform the robotic puppet show

Title	Perform the robotic puppet show.
Description	As the screenplay detail, all of the robotic puppets, sound effect, and background music play synchronously.
Actors	Directors and robotic puppets.
Pre-condition	<u>Desktop:</u> <ol style="list-style-type: none"> 1. The device has the ability to access the website by using a web browser. 2. Install the lite-server (Node.js). 3. The device and robotic puppets need to be in LAN. <u>Mobile:</u> <ol style="list-style-type: none"> 1. Download the Android application and install. 2. The mobile device and robotic puppets need to be in LAN.
Post-condition	All of the robotic puppets, sound effect, and background music start playing as the screenplay.
Scenario	<ol style="list-style-type: none"> 1. Log in the website. 2. Select the name of a screenplay. 3. Set up the each puppet's IP address and port number. 4. Click the "start" button to perform the show.

Table 3-18 Use case of pick different paths as the story progresses

Title	Pick different paths as the story progresses.
Description	<ol style="list-style-type: none"> 1. When the story has branching, the Node.js will send a command to one of the DARwIn-OP in order to detect the color presented by the user 2. The Node.js selects a child node in the scene tree to perform.
Actors	Directors and robotic puppets.

Pre-condition	The mother node that has more than two children has been completed.
Post-condition	The child node will be chosen and performed according to the result of color detection.
Scenario	<ol style="list-style-type: none"> 1. Log into the website. 2. Select a screenplay that has branching to play. 3. While branching, the user presents the color in front of DARwIn-OP's USB camera. 4. The color scene that was selected will be performed.



Figure 3-5 Use case diagram of EPFS

3.3 Motion Generation

3.3.1 Preprogrammed Robotic Motion

The robotic puppets such as DARwIn-OP and NAO have provided user with some applications for rapidly developing a variety of postures and motions stored within the robot which will be applied to robotic puppet shows then. Below are development tools for DARwIn-OP and NAO.

(1) RoboPlus for DARwIn-OP

The official software is called the “RoboPlus motion”, this software allows users to edit the continuous motion of a robot which is saved as a page in a fast and simple way as shown in Figure 3-6. Each page contains the information for one single motion including: the duration time for each step, the speed rate and value of the actuators, and the number of repetitions needed to complete a full motion. By doing so, the real execution time can be precisely calculated; this in turn helps the director arrange the screenplay.

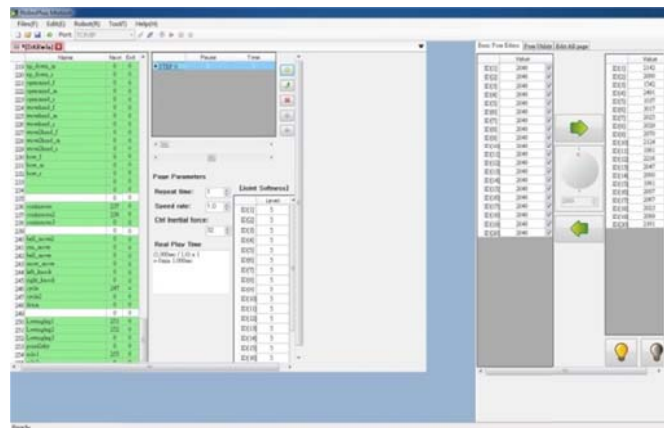


Figure 3-6 RoboPlus interface

(2) Choregraphe for NAO

Choregraphe is software developed for the NAO robot, which gives users the ability to control the robot by dragging or customizing a variety of event boxes within the GUI as shown in Figure 3-7. Using Choregraphe can allow the developer easily observe the angles amongst each joint in NAO so that the motion can be programmed as a file in Python and recorded within the robot. By using this tool, user can compose a variety of different motions for performances in an efficient way.

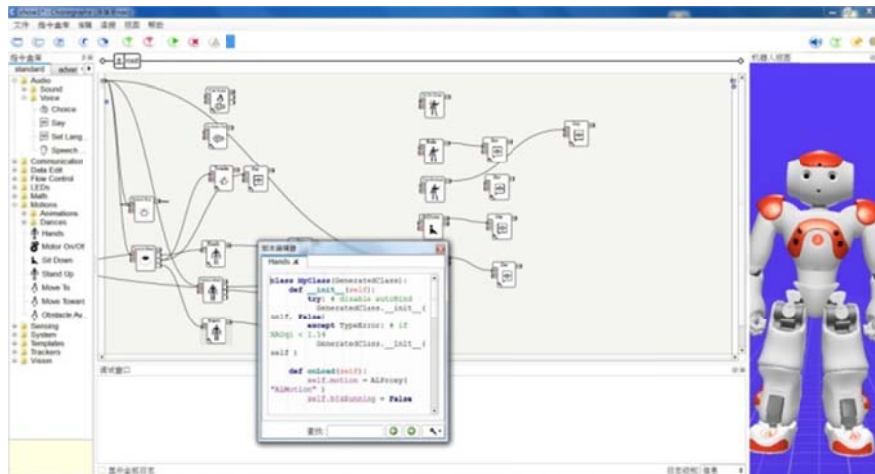


Figure 3-7 Choregraphe interface

After those motion pages or files have been created and stored within the robots, the director can use the motion management page in ISAP to add those motions into the platform and then apply them to robotic puppet shows.

3.3.2 Kinect-based Motion Uploading

This platform also allows users to utilize Kinect to capture human motions for extending the motion library applied to robotic puppet shows. By doing so, motion generation can become more intuitive and efficient than the common ways used to control robots. Below are the methods proposed by [7] to implement motion emulation and representation by using Kinect.

When the human's image is successfully captured by Kinect, a set of lines that mimic a skeleton will then appear on the top of the body. After the skeleton is constructed, the user can receive important data such as joint position and orientation.

The flow chart of skeleton tracking algorithm is shown in Figure 3-8.

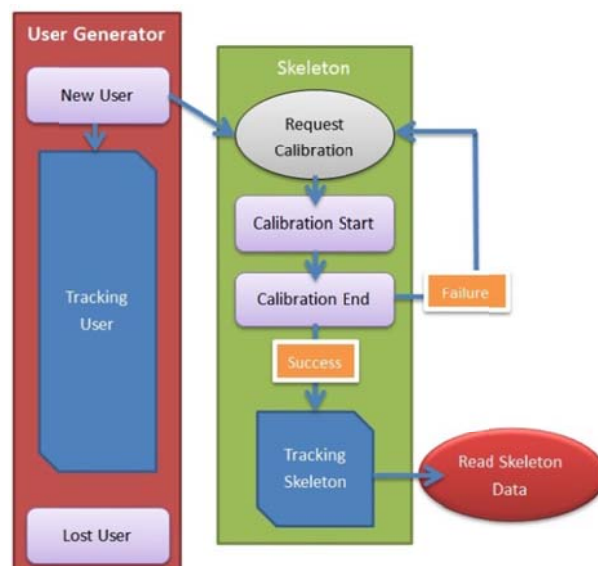


Figure 3-8 NITE skeleton tracing algorithm

Kinect provides a parameter comprised of a joint orientation that is composed by using a 3 x 3 orthogonal unit matrix. X / Y / Z are the three orientations that represent the posture of the current joint. In OpenNI, the definition of orientation is as follows, “A joint orientation is described by its actual rotation and the confidence we have in that rotation” .

- The first column is the X orientation, where the value increases from left to right.
- The second column is the Y orientation, where the value increases from bottom to top.
- The third column is the Z orientation, where the value increases from near to far.

Inverse kinematics refers to the use of a robots kinematics equations to determine the joint parameters that provide the desired position of the end-effector. The joint orientation represents current posture. Solving the inverse kinematics can determine the current joints change in angle. First, the relationship between the coordinate system of each joint must be defined. Kinect's infrared lens is the origin of the coordinates. The unit used for three-dimensional space in a right-handed coordinate system is millimeters.

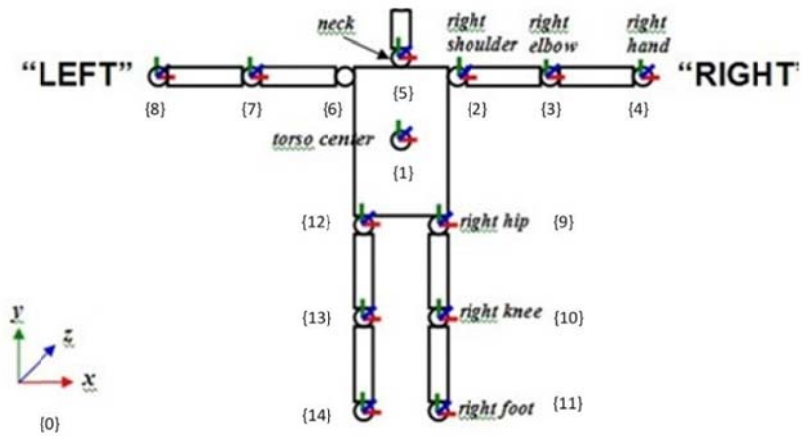


Figure 3-9 Coordinate relationship diagram

According to the Figure 3-9, the relationship of the coordinate system is defined as follows, and because of the balance problem, this thesis only considers the coordinates located in the upper body of the skeleton.

$Rot(Z, \theta_z)$ is the homogeneous rotation matrix which rotated θ_z around the Z axis.

$Rot(Y, \theta_y)$ is the homogeneous rotation matrix which rotated θ_y around the Y axis.

$Rot(X, \theta_x)$ is the homogeneous rotation matrix which rotated θ_x around the X axis.

If the joint coordinate is belong $\{i\}$, then it's joint position is (X_i, Y_i, Z_i) . For example the $\{0\}$ is the origin of system coordinate. The position is (X_0, Y_0, Z_0) and $X_0 = 0, Y_0 = 0, Z_0 = 0$.

$$\{0\} \rightarrow \{1\} = {}^0_1T = Rot(Z, \theta_{z1}) \cdot Rot(Y, \theta_{y1}) \cdot Rot(X, \theta_{x1}) \cdot Trans(x_1 - x_0, y_1 - y_0, z_1 - z_0) =$$

$$\begin{bmatrix} C\theta_{z1} & -S\theta_{z1} & 0 & 0 \\ S\theta_{z1} & C\theta_{z1} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C\theta_{y1} & 0 & S\theta_{y1} & 0 \\ 0 & 1 & 0 & 0 \\ -S\theta_{y1} & 0 & C\theta_{y1} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\theta_{x1} & -S\theta_{x1} & 0 \\ 0 & S\theta_{x1} & C\theta_{x1} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & X_1 - X_0 \\ 0 & 1 & 0 & Y_1 - Y_0 \\ 0 & 0 & 1 & Z_1 - Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\{1\} \rightarrow \{2\} = {}^1_2T = Rot(Z, \theta_{z2}) \cdot Rot(Y, \theta_{y2}) \cdot Rot(X, \theta_{x2}) \cdot Trans(x_2 - x_1, y_2 - y_1, 0) =$$

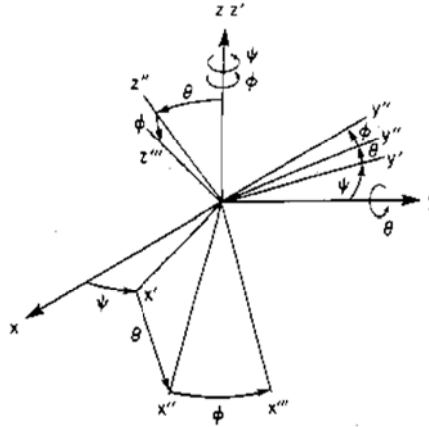


Figure 3-10 PRY convention

In Figure 3-10, RPY (Roll Pitch Yaw) convention in the inverse- kinematic transformation matrix is used to calculate the postures angle change. The original joint orientation is a 3×3 orthogonal matrix, the matrix needs to be converted to homogeneous and then continue computing. The function is as follows. T_{OH} is a homogeneous transform matrix which represents the joint posture.

$$T_{OH} = \text{Rot}(Z, \theta_z) \cdot \text{Rot}(Y, \theta_y) \cdot \text{Rot}(X, \theta_x) = \begin{bmatrix} nx & ox & ax & 0 \\ ny & oy & ay & 0 \\ nz & oz & az & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The coordinate $\{i\}$ can be represented as the following

$${}^0T_i = {}^0T_1 \cdot {}^1T_2 \dots {}^{i-2}T_{i-1} \cdot {}^{i-1}T_i = \begin{bmatrix} nx & ox & ax & px \\ ny & oy & ay & py \\ nz & oz & az & pz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Then

$${}^0T_i = {}^0T_1 \cdot {}^1T_2 \dots {}^{i-2}T_{i-1} \cdot {}^{i-1}T_i = {}_{i-1}^0T_i {}_i^1T_{i-1}$$

$$({}_{i-1}^0T_i)^{-1} \cdot {}^0T_i = {}_i^1T_{i-1}$$

The actual rotation angles of joints when solving for the {i} coordinate and the inverse-matrix of {i-1} coordinate which are relative to the {0} coordinate system. To solve the RPY, $\text{Rot}^{-1}(Z, \theta_z)$ must be multiplied in both sides of the equation.

$$\text{Rot}^{-1}(Z, \theta_z) \cdot \text{TOH} = \text{Rot}(Y, \theta_y) \cdot \text{Rot}(X, \theta_x)$$

$$\begin{bmatrix} C\theta_z & S\theta_z & 0 & 0 \\ -S\theta_z & C\theta_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} nx & ox & ax & 0 \\ ny & oy & ay & 0 \\ nz & oz & az & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} C\theta_y & 0 & S\theta_y & 0 \\ 0 & 1 & 0 & 0 \\ -S\theta_y & 0 & C\theta_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\theta_x & -S\theta_x & 0 \\ 0 & S\theta_x & C\theta_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Simplify the left and right matrix

$$\begin{bmatrix} nx C \theta_z + ny S \theta_z & C \theta_z \cdot ox + S \theta_z \cdot oy & C \theta_z \cdot ax + S \theta_z \cdot ay & 0 \\ ny \cdot C \theta_z - nx \cdot S \theta_z & -S \theta_z \cdot ox + C \theta_z \cdot oy & -S \theta_z \cdot ax + C \theta_z \cdot ay & 0 \\ nz & -oz & -az & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} C \theta_y & S \theta_y \cdot S \theta_x & S \theta_y \cdot C \theta_x & 0 \\ 0 & C \theta_x & -S \theta_x & 0 \\ -S \theta_y & C \theta_y \cdot S \theta_x & C \theta_y \cdot C \theta_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

And the result is as follows

$$ny \cdot C \theta_z - nx \cdot S \theta_z = 0$$

$$\theta_z = \text{ATAN2}(ny/nx)$$

$$\begin{cases} C \theta_y = nx \cdot C \theta_z + ny \cdot S \theta_z \\ -S \theta_y = nz \end{cases}$$

$$\theta_y = \text{ATAN2}(-nz/(nx C \theta_z + ny S \theta_z))$$

$$\begin{cases} C \theta_x = -S \theta_z \cdot ox + C \theta_z \cdot oy \\ -S \theta_x = -S \theta_z \cdot ax + C \theta_z \cdot ay \end{cases}$$

$$\theta_x = \text{ATAN2}((ax \cdot S \theta_z - ay \cdot C \theta_z)/(oy \cdot C \theta_z - ox \cdot S \theta_z))$$

According to the formula above, actual rotation angles can be calculated for each actuator. There are twenty MX-28 actuators in DARwIn-OP that give the robot the

ability to move its joints. In order to calculate how to reach the desired position, the change in angles must be translated into units. In addition, there are 4096 units within every 360 degree rotation, each unit represents 0.088 ($360/4096$) degrees.

When the movements have been recorded they need to be automatically saved in the system for future use. There are large quantities of data saved due to the fast speed of data generation during the movement capture process. Because of this, key-poses [7] are separated from the redundant postures that take up resources. The key-pose selection process is described below:

Step1: The rotation of the joints is determined by using Inverse Kinematic

Step2: If the change in angles exceeds the threshold, then the state is recorded as an action; if it does not, then the state is considered to be static.

Step3: If the current state differs from the state it follows, then the key_pose_bit will be equal to 1, otherwise it will be equal to 0.

Step4: If the key_pose_bit is equal to 0, then the previous pose will be checked to see whether the direction of movement has changed. If it has, the key_pose_bit will be equal to 1, otherwise it will remain equal to 0.

Step5: Saving the current data into the old data buffer.

The flow chart of steps above can be drawn in Figure 3-11.

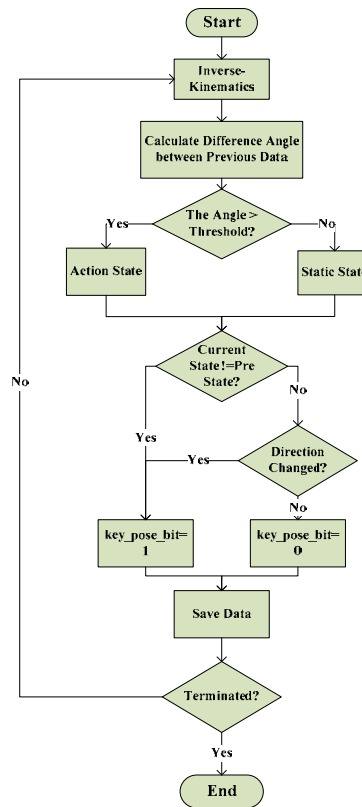


Figure 3-11 Key-pose selection flow chart

When a new pose is captured the time will be recorded and the difference in time between the two key-poses will be calculated in order to obtain the execution time. The execution time will be stored in the time_buffer and used to calculate the speed of the actuator. The presence or absence of continuous motion between the current and next key-pose will be judged by the continuous_bit.

The continuous_bit is designed to determine whether an actuator is in continuous motion or not. If the key-pose is in continuous motion then the continuous_bit will recognize it as a one, this means that while the actuator is moving to the continuous key-pose the following pose will already begin to load. With the assistance of continuous_bit the representation of the key-poses will be smoother.

If the previous pose is regarded as a key-pose then the system will check if any of the actuators are still in motion. If there is more than one actuator still in motion, the continuous_bit will be recorded as 1 which represents that the previous posture is continuous. In the case that all actuators are static, then the continuous_bit will be recorded as 0.

The packet constructed by the methods mentioned above contains the actuator's position, speed, continuous_bit, and execution time from the key-pose, as shown in Figure 3-12. Those kinds of packets are used as postures within Kinect-based motion files for robotic motion representation. As a result, the Kinect-based motion files can be uploaded to ISAP in order to extend the robotic motion library.

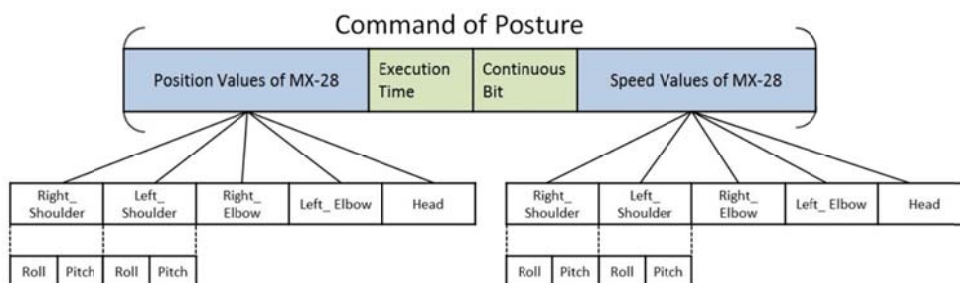


Figure 3-12 Posture within Kinect-based motion file

3.4 Screenplay Editing

3.4.1 Scene Tree

A rooted n-ary tree structure is used to construct a whole screenplay because this kind of tree allows each node to diverge a maximum of n times. By using this kind of tree, the users make the plot of story diverge and as a result, the story will become

more diverse. There are three different crucial nodes in the tree: root node, decision nodes, and leaf nodes. Each node stands for a single scene, which was depicted in Figure 3-13. The root node represents the start of the story. Any node that has more than two child nodes is called a decision node. Branching will occur when the first node has finished performing, and then the user can choose one of scenes at the next level of the tree to make the story diverge and proceed. As a result, the leaf node means the end scene of the screenplay. By doing so, even the user play the same screenplay, the development and ending of the story can be distinctive according to the user's choice.

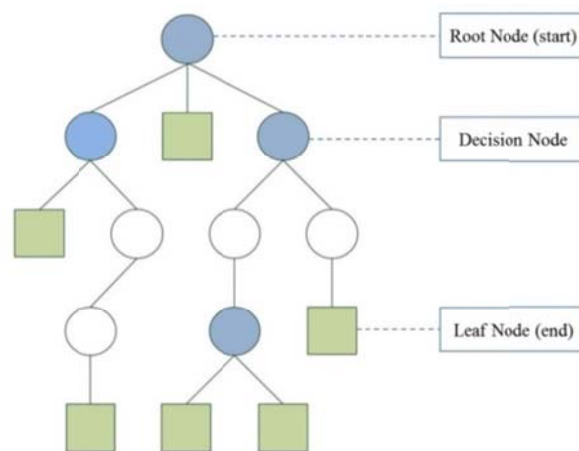


Figure 3-13 Structure of scene tree

3.4.2 Time Stamp

Compared to event-driven story arrangements, the time stamp feature provides the director with a precise way to arrange the content of the performance. Doing so makes the robotic puppet show more lively and attractive. The feature can be applied to the two parts below.

(1) Expression in Motion and Speech

In robotic puppet shows, the motion and speech ability are needed to express the actors emotions. Therefore, the fact that the motion and speech of each robot starts at the same time or overlaps in order to fit the director's requirements has played an essential part in the scene editing process.

(2) Interaction between Robots and Music

Like real dramas, robotic puppet shows have a cast of actors that perform on stage. The task of arranging and recording the interaction between puppets requires sophisticated controls. In addition, EPFS has joined sound effects and background music to increase dramatic tension.

In order to solve the expression and interaction above, the timeline (which is based on seconds) is built within each scene as shown in Figure 3-14. Each puppet, music, and sound effect has its own track within the timeline that can record performance details such as motion, speech, sound effects and background music. The robotic puppet track has two sub tracks used to individually record the content of motion and speech. GUI makes the color boxes that represent the different performance details.

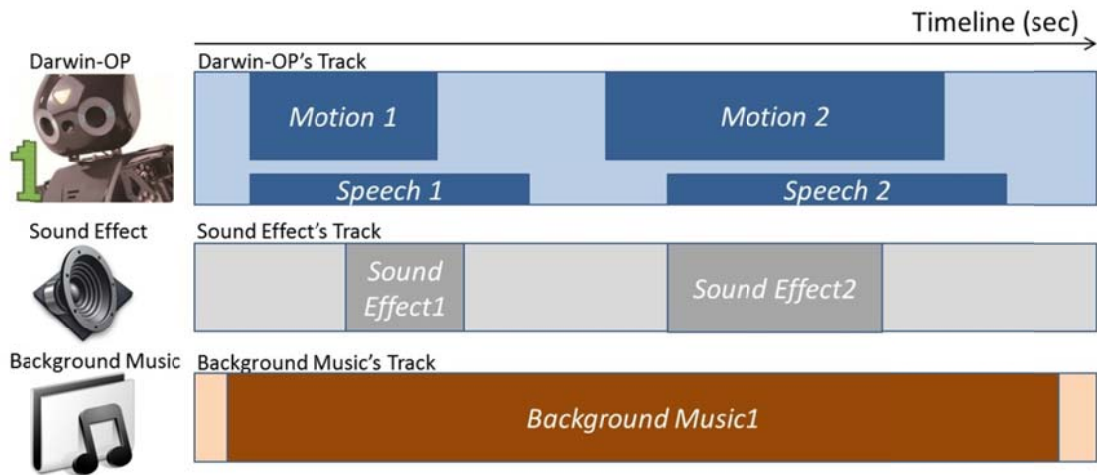


Figure 3-14 Timeline tracks within a scene

3.4.3 Screenplay Format

The JSON (JavaScript Object Notation), a subset of JavaScript syntax, is a text format that is mainly used to exchange data. It is straight forward enough to be understood and used by computers as well as people. Figure 3-15 [12] shows that JSON is built on two structures: object and array.

The name/value can be found in between the left right bracket of the object. Colons are used to distinguish the names from the values and each set of name/values is broken apart by a comma. On the contract, the array is a sequential data grouping. Brackets are placed on both ends and commas are used to distinguish between values. There can be different types of values within an array such as number, string, object, and array.

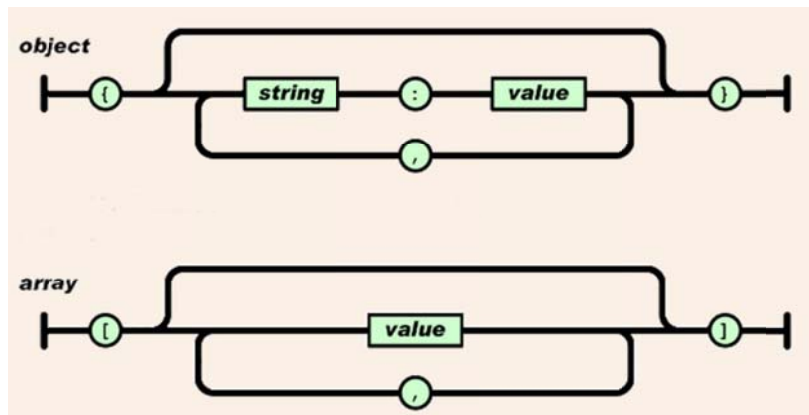


Figure 3-15 JSON forms

JSON and XML (Extensible Markup Language) are designed to describe structured data for data interchange purposes. Compared to XML, the data encoded in JSON is simpler, and easier for humans to read and write. In addition, the size of JSON data is smaller than an encoding containing an equivalent amount of information in XML. While parsing using a web browser, JSON is faster than XML. The JSON was chosen as the format of the screenplay to be used with the ISAP based web application.

When the user has selected and performed a screenplay with a web browser, the JSON data which records the details of this screenplay will be automatically generated from the relational database. The JSON data is formatted as shown in Figure 3-16 and Figure 3-17.

3.5 Screenplay Performance

This thesis, An Editorial Platform for Screenplay of Interactive Robotic Puppet Shows (EPFS) is made up of two parts: the interactive screenplay authoring platform (ISAP) and the mobile interpreter (MI), as shown in Figure 3-18.

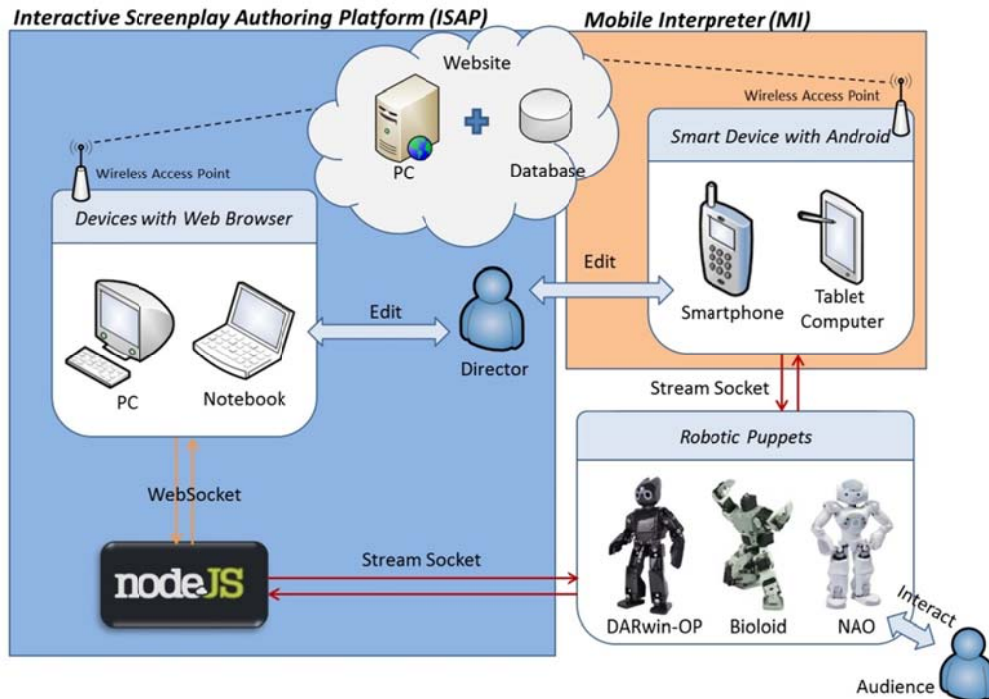


Figure 3-18 EPFS architecture

3.5.1 Interactive Screenplay Authoring Platform (ISAP)

ISAP contains two parts: the website and the Node.js. This website was chosen because it allows easy-access and friendly interface for users of different ages. The main function of this website is to provide users with screenplay display, editing, and storage.

The implementation of website follows MVC framework, which includes Model, View and Controller as shown in Figure 3-19. Model is only in charge of data

structure from the rational database. View provides users with the visual interface of the website by using HTML, CSS and JavaScript. Controller is the bridge between Model and View, which is responsible for business logic and data control. Users can access the website by using any device with a web browser to edit their own screenplays. All of the details will be automatically stored in web server within database.

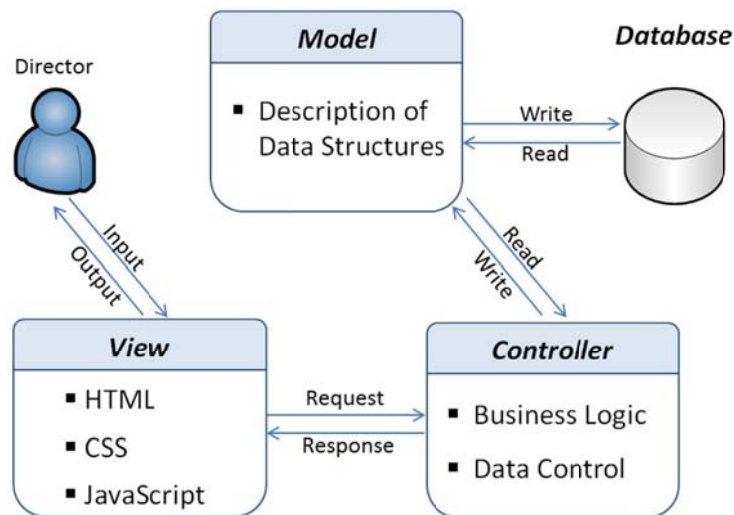


Figure 3-19 Website based on MVC

A virtual IP (VIP) is assigned to the device with web browser and robotic puppets when they are connected to the local network via Wi-Fi. In order to deliver the packet of the screenplay to robotic puppets that support VIP, The lite server, Node.js, is built between the web browser and robotic puppets as depicted in Figure 3-20. Node.js is a server-side scripting language based on JavaScript. It is designed to easily develop network applications.

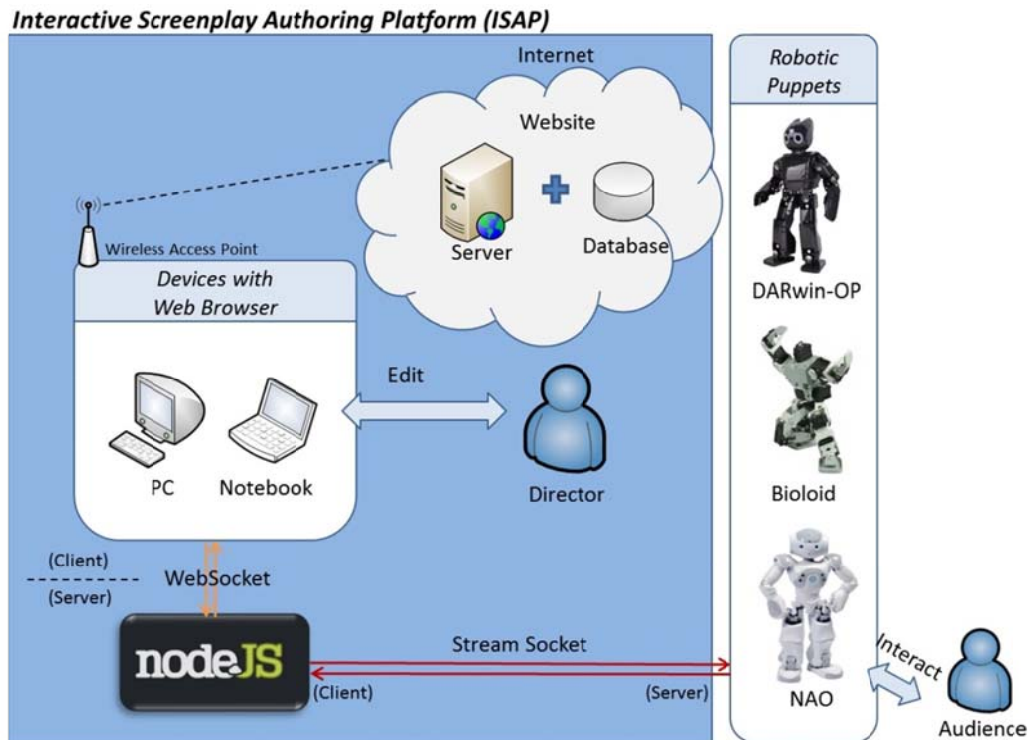


Figure 3-20 ISAP architecture

Node.js is a platform using V8 JavaScript Engine to utilize JavaScript scripts without web browser. It also provides developers lots of APIs to build the real-time network application. In spite of the fact that Node.js is single threaded, the event loop and non-blocking operations allow users to implement the parallel processing [13].

With the assistance of Node.js in ISAP, the two methods used for communication are WebSocket and Stream Socket. “WebSocket is defined as a full-duplex single socket connection between client and server for data exchange [14].

Once established, the WebSocket data frames can be sent back and forth between the client (web browser) and the server (Node.js) in ISAP. In comparison to the polling, the WebSocket establishes a continuous connection to increase the data transfer rate

and reduce network latency, as shown in Figure 3-21 [14].

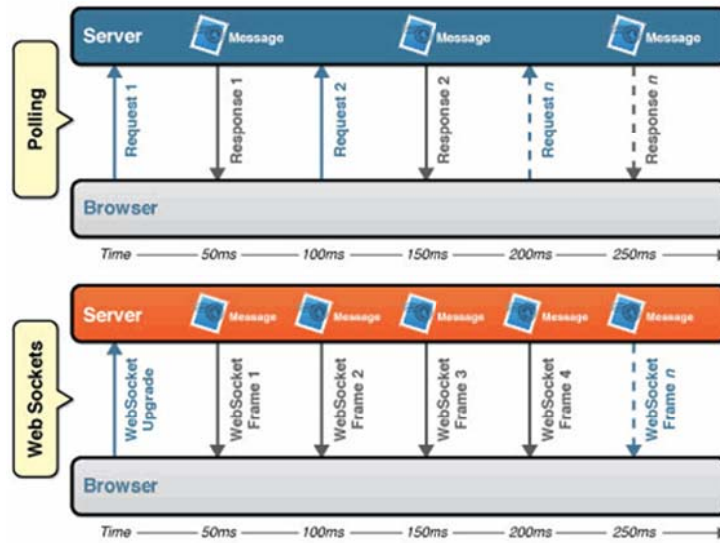


Figure 3-21 Polling and WebSocket latency comparison

On the other side, Stream sockets provide a bidirectional channel using TCP/IP protocol to guarantee the clients receive the correct and unrepeated data in proper order [15]. In this platform, the stream sockets are created to implement a channel between the client (Node.js) and the servers (robotic puppets).

First, the Node.js application should be run as a WebSocket server in the background in order to start listening to the client-side web browser. Once the director has selected a screenplay to play and has entered into the settings page with a web browser, the WebSocket will be connected. After all of the settings including the IP addresses and port numbers have been filled in and submitted by the director, the browser will send a request to the web server for fetching the designated screenplay that was formatted using JSON as shown in Figure 3-22.

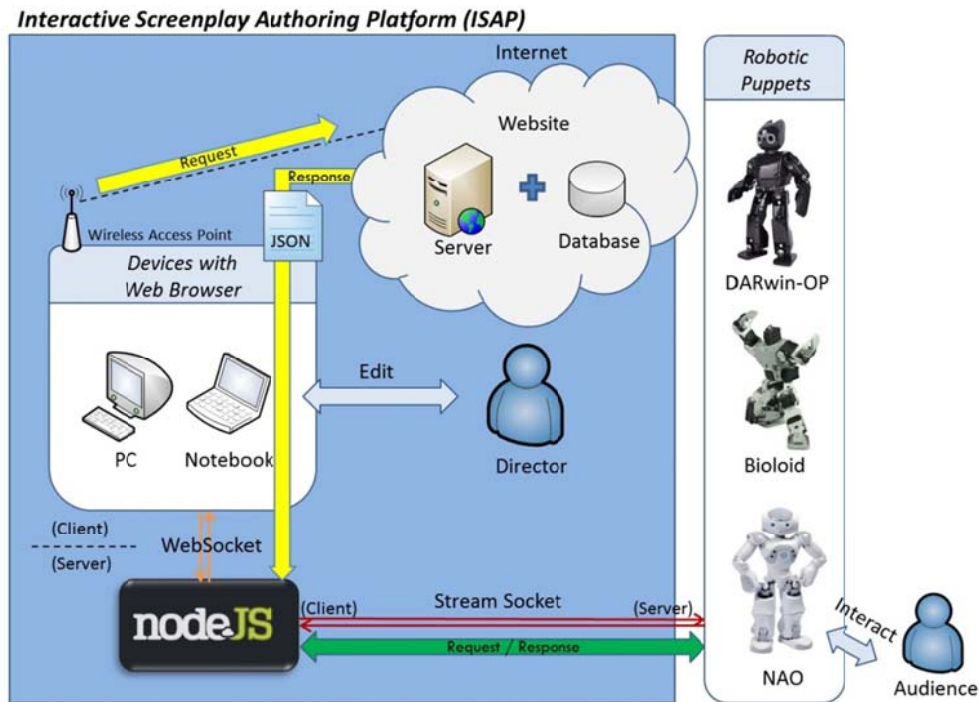


Figure 3-22 The process of performing a screenplay

Next, the JSON file will be delivered to Node.js server via WebSocket and interpreted level by level to read the settings and nodes object as shown in Figure 3-22 within this screenplay. In this step, the Node.js plays not only in the WebSocket server but also in the stream socket client in order to connect to each robotic puppet according to specific IP addresses and port numbers recorded in JSON file.

Figure 3-23 shows after the stream sockets have been created, the Node.js will pick up the root scene, all of commands that belong to the timeline item will be set up to be executed according to its designated delay time. Moreover, if the commands belong to the robotic puppets, they will be taken into the socket array and sent to each robotic puppet through the stream socket. The sound effects and music commands will be played by calling the external media software.

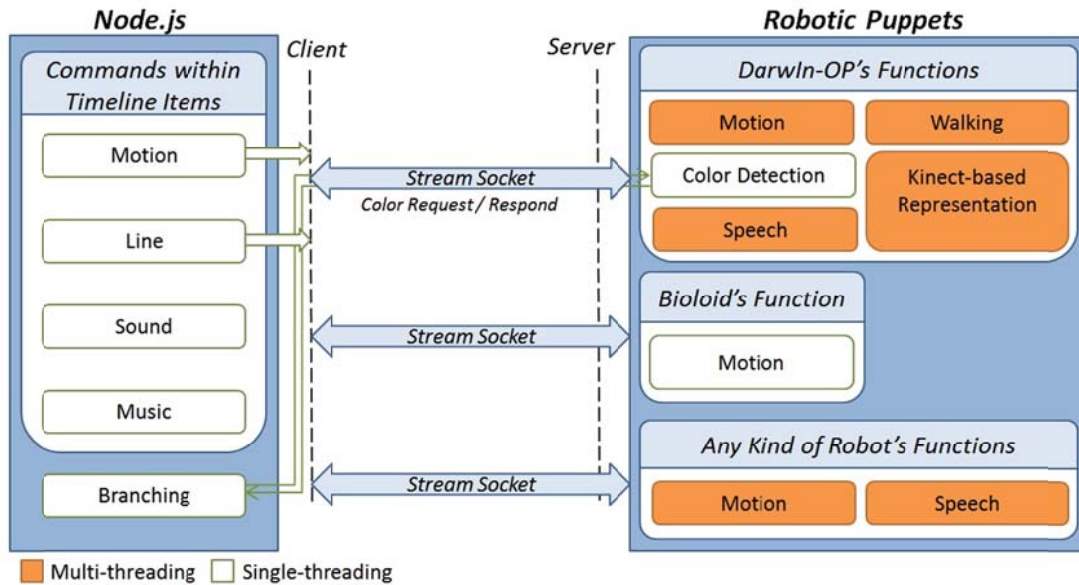


Figure 3-23 Command execution by Node.js via stream sockets

When the present scene has been finished, Node.js will check whether the scene has children. If it does, the children scene will be performed. Scenes that have more than two children will cause branching to occur as shown in Figure 3-24. In that case, Node.js will send a request to the DARwIn-OP robot to run the color-detection function. After the user presents the color, the result will be send back to Node.js in order to select and perform the corresponding color children scene. If the color distinction has failed three times, DARwIn-OP will return a signal to make Node.js pick up one of the existing children randomly in order to guarantee the performance process can continue. The whole process is illustrated in Figure 3-25.

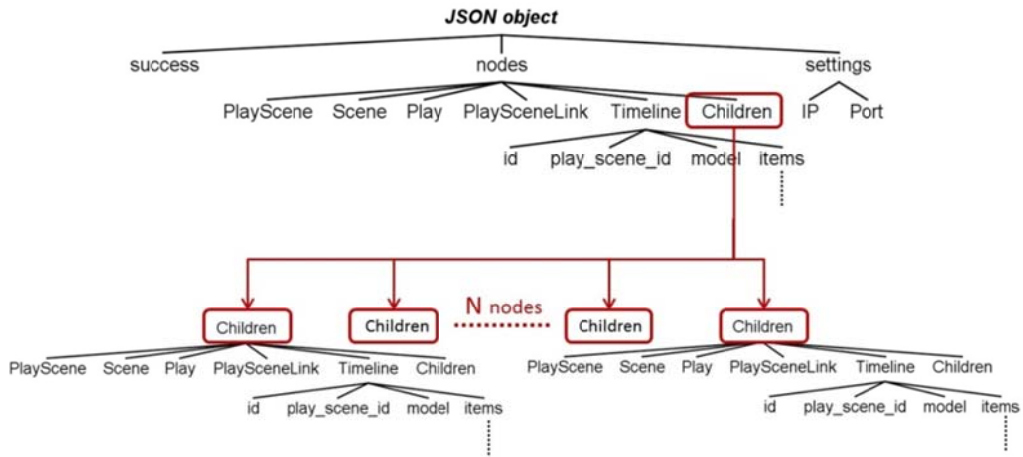


Figure 3-24 Multiple child nodes cause branching

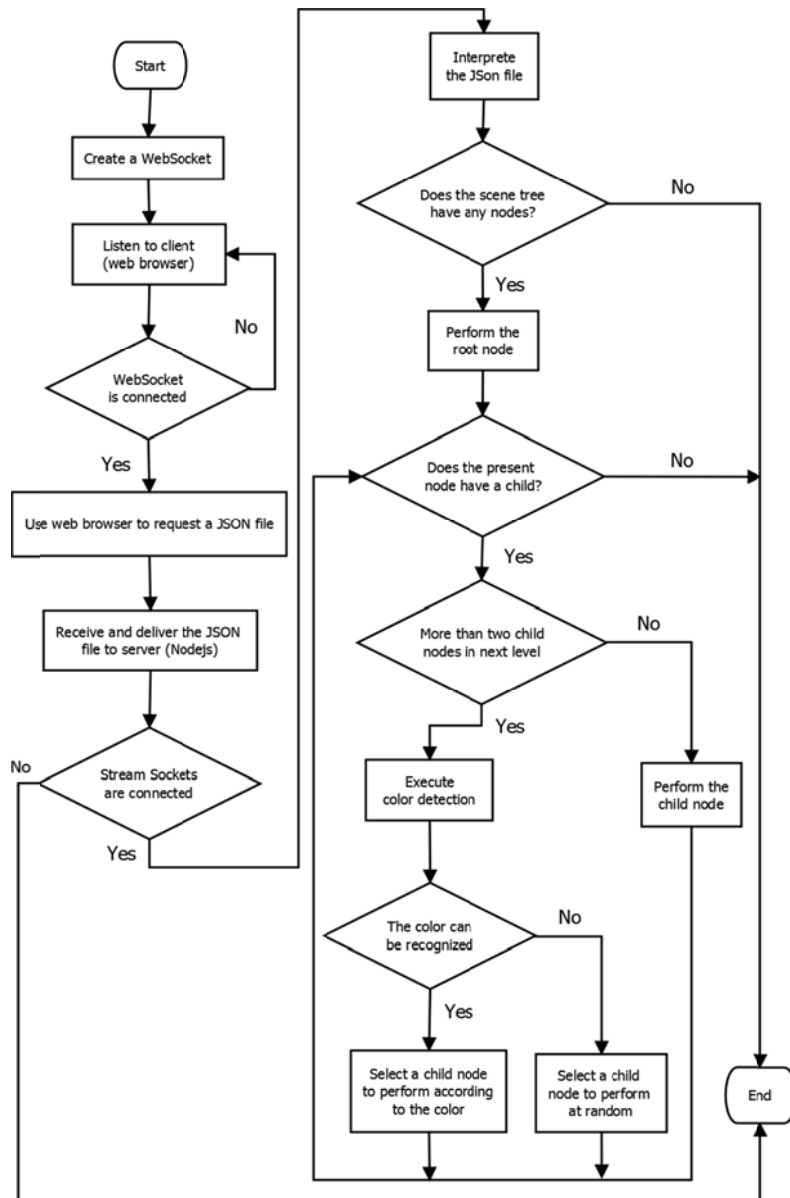


Figure 3-25 Node.js process

In order to translate different commands to specific robotic puppets through stream sockets, EPFS has designed three different types of packets. Furthermore, when the robots have received the packets, the command within each packet will be extracted and executed by the related functions. The command with the type I packet represents the motion page, line or file name defined by users. On the contrary, the type II and III packets contain the executable instructions.

(1) Type I

As shown in Figure 3-26. The first character in the packet is used as the identifier, which allows DARwIn-OP to tell whether the command is applied to the motion, walking or speech function. And the content of the command is attached to the remaining part of the packet after the first character.

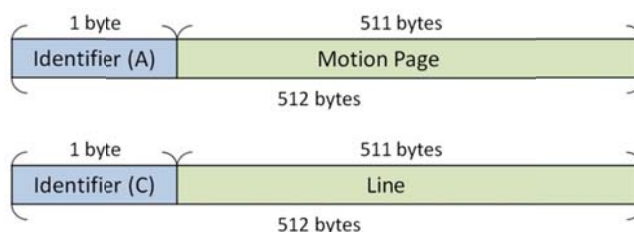


Figure 3-26 The format of the type I packet

In addition, EPFS allows users to extend any kind of robotic puppet into the platform for motion and speech arrangement. The new robot's packet should follow DARwIn-OP's format by using the first identifier character, and the remaining part of the packet will contain the new robot's motion and line command. In this platform, the NAO robot is regarded as a new addition to the performance.

(2) Type II

The Kinect-based motion files can be uploaded to the web server and performed by DARwIn-OP. Each posture that is extracted from a motion file will be individually written into a packet in sequence. Each packet is similar to DARwIn-OP because the first identifier character is used and the remaining part stores the detailed commands of the posture. Figure 3-27 shows that the first packet is designed by using the character “N” and the amount of postures within remaining part of the buffer. The other packet stores the detailed posture data with first character “B”.

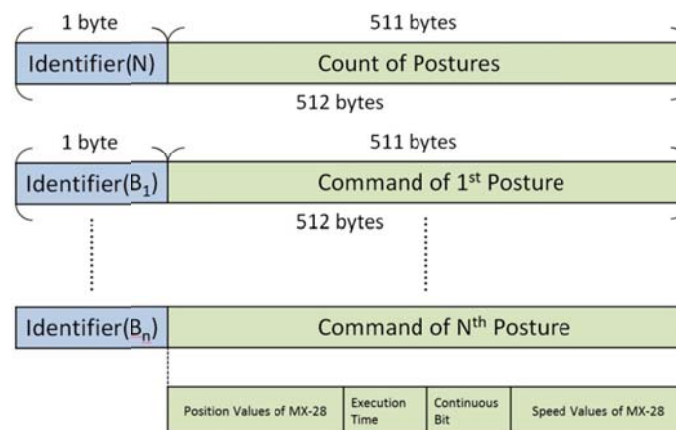


Figure 3-27 The format of the type II packet

(3) Type III

Figure 3-28 shows the packet that is designed to contain a single posture for controlling Bioloid robots. The structure of the command packet follows the AX-12 actuator datasheet [16]. In this thesis, the 128 byte length packet consists of one byte blocks such as 0XFF, 0XFF, Broadcasting ID, LENGTH, INSTRUCTION, and CHECKSUM. The only exception is the PARAMETERS block which was calculated

as having 120 bytes. In order to use the Synch Write instruction to control multiple actuators at the same time, the INSTRUCTION block must be set to 0X83 and the Broadcasting ID block to 0XFE [16].

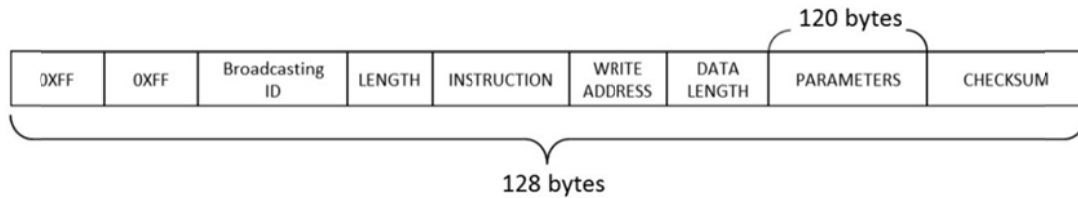


Figure 3-28 The format of the type III packet

3.5.2 Mobile Interpreter (MI)

In order to make the robotic puppet show more portable and easier to perform, an Android application has been developed as a mobile interpreter (MI). This application can be run on various Android devices such as a tablet or a smartphone. Based on the existing MVC framework, this interpreter can be regarded as a new View constructed with the original Model and Controller. The benefit of using MVC framework is that the separation of the patterns allows for the development be finished in a clear and efficient way. Compared to ISAP, the MI is playing the role that is similar to the combination of devices within a web browser and Node.js as show in Figures 3-29. The major difference between ISAP and MI is that the latter one only provides basic functions including screenplay display and performance.

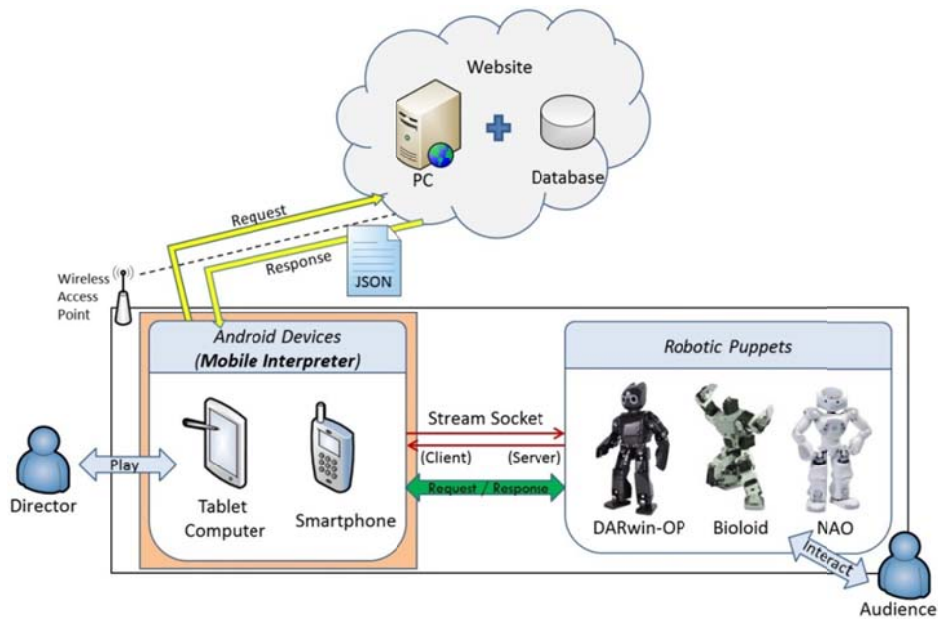


Figure 3-29 MI architecture

The Android device and the robotic puppets are required to connect to a wireless LAN while performing. When the director has logged into the MI, the application will update the list of screenplays from the web server via Internet. Once the director has selected a screenplay to perform, the application will ask the web server again to request the corresponding JSON formatted screenplay which is identical to the one found in ISAP. Like the functions belonging to Node.js, the MI also has the ability to create stream sockets to communicate with each robotic puppet. The JSON file will be interpreted into different commands and be sent to designated actors at specified times. Furthermore, the sound effects and background music will be played by a built-in function according to the file name of the music commands. While branching, the MI can pick different story paths based on the results of the color detection process completed by DARwIn-OP through the stream socket.

3.6 Hardware Architecture

For drama performances, the DARwIn-OP Bioloid and NAO robots are chosen as the actors used in the EPFS because of their human-like appearance and biped-walking features. Furthermore, both of them are supported by a Wi-Fi wireless network that uses an open source character which allows programmers to easily develop a given robots basic motions or even various new functions. Therefore these robots are used in the fields of academic research, intelligent robotic competition and entertainment.

3.6.1 DARwIn-OP

As shown in Figure 3-30 [17], DARwIn-OP is equipped with twenty Dynamixel MX-28 servo actuators, a USB camera, MIC, speaker, and multiple I/O devices that include HDMI and USB ports. The DARwIn-OP has the ability to operate in real time, and it also comes pre-installed with Ubuntu 9.04 operating system.

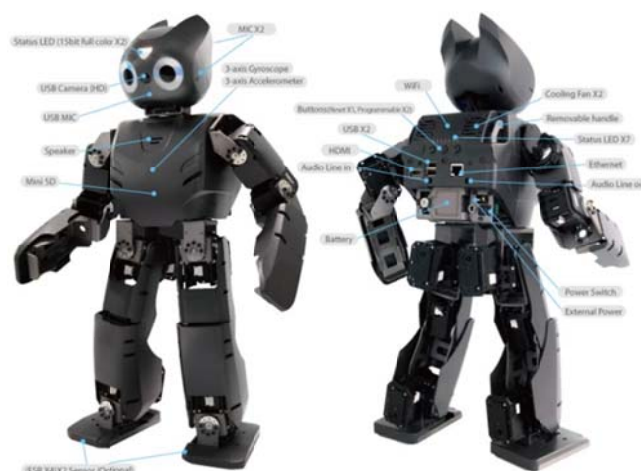


Figure 3-30 DARwIn-OP hardware

3.6.2 Bioloid

The Bioloid consists of eighteen Dynamixel AX-12 servo actuators and unfixed units, which are shown in Figure 3-31. The name Bioloid was chosen in the spirit of that all of the main mechanisms of biological entities can be separated and reassembled as depicted in Figure 3-32. This feature allows users to have the ability to design their own robots that can adapt to a variety of different environments. For instance, snake robots have the ability to climb poles and spider robots are able to trek across uneven ground.



Figure 3-31 Front and rear view of Bioloid



Figure 3-32 Bioloid applications

In order to connect to the Wi-Fi network, the USR-WIFI232-2 module was chosen to as a controller that was attached to the Bioloid's back, as seen in Figure 3-33. The module enables the Bioloid to perform actions depending on the instructions which have been sent from the platform and transmitted to its AX-12 actuators via TX pin. There are two settings for USR-WIFI232-2, AP mode and Station mode.



Figure 3-33 Bioloid equipped with USR-WIFI232-2 module

3.6.3 NAO

NAO is highly efficient, programmable humanoid robot developed by Aldebaran Robotics. It has been widely used in fields such as research, education and entertainment. NAO robots have 25 degrees of freedom, and a variety of visual, audio, language, movement coordination, and tactile sensors as shown in Figure 3-34 [18].

The sensors mentioned above and the strong, efficient design of the robot's body both contributed developers the ability to implement creative, powerful and reliable applications based on NAO.

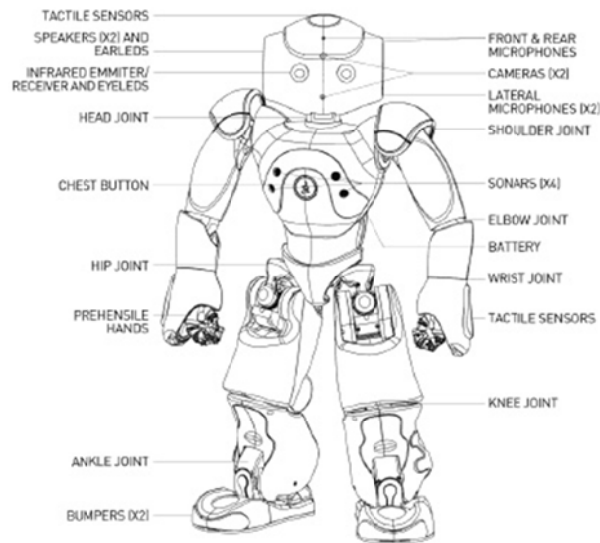


Figure 3-34 Hardware framework of NAO

3.6.4 Smart Device

The smart and handheld device has become popular for daily human application. Devices such as smartphones and tablets are usually equipped with a variety of useful tools like cameras, touch panels, Wi-Fi, accelerometers and gyroscopes. MI is designed to allow users of all ages to play the screenplay for robotic puppet shows in a fast and simple way. Therefore, the large size of the panel provides users with a better experience while reviewing the list of screenplays and operating the robotic puppet show. The operating system running on the devices plays an essential role for application development and usage. Compared to the iOS, Android has less restrictions and more market share since its features like free of charge, open source and high degree-of-freedom.

In addition, the device should support Wi-Fi in order to create the stream sockets with robotic puppets. An mp3 encoder is also needed to play the music applied to the sound effects and background. For reasons outlined above, the ASUS MEMO Pad FHD 10 seen in Figure 3-35 has been selected as the mobile controller for the robots.

Table 3-19 shows the main features of MEMO Pad FHD 10.



Figure 3-35 Appearance of ASUS MEMO Pad FHD 10

Table 3-19 The ASUS MEMO Pad FHD 10 product specification

Function	Description
Operating System	Android 4.2
Display	10.1" LED Backlight WUXGA (1920x1200) Screen
CPU	Intel® Atom™ Z2560 Dual-Core, 1.6 GHz
Memory	2GB
Graphic	PowerVR SGX 544MP
Wireless Data Network	WLAN802.11 a/b/g/n*1 Bluetooth V3.0
Camera	1.2 MP Front Camera 5 MP Rear Camera with Auto focus
Audio	Stereo Speakers with SonicMaster technology
Interface	2-in-1 Audio Jack (Head Phone / Mic-in) 1 x Micro USB 1 x Micro HDMI 3 x Micro SD Card Reader

3.7 Software Architecture

3.7.1 DARwIn-OP API

DARwIn-OP is initially installed with the Linux-like operating system named Ubuntu, which is a stable, easy to maintain, open-source software. The functions of the DARwIn-OP robot can be divided into two modes, multi-threading and single-threading. The multi-threading mode includes motion, walking, Kinect-based representation and speech functions. On the other hand, color detection is developed in single-threading. POSIX (Portable Operating System Interface for UNIX) defines a standard threading library named *Pthread* which can be used for solving the lip synchronization problem. In multi-threading mode, the *Pthread* library allows for four functions to be run simultaneously. There are six steps to operating a thread, as depicted in Figure 3-36. Step1 is to declare the ID and attribute of the thread, then its attribute will be set to default values in step2. Most importantly, the attribute of thread must be reset by calling the function named *pthread_attr_setdetachstate()*. There are two parameters, the attribute address and attribute value, both of them can be put into the function. The value of the attribute can be divided into two parts : *PTHREAD_CREATE_JOINABLE* and *PTHREAD_CREATE_DETACHED*.

PTHREAD_CREATE_JOINABLE represents the thread and is created in a joinable state, which keeps its resource until other threads have been destroyed. On

the contrary, *PTHREAD_CREATE_DETACHED* allows the thread to be created in a detached state, which will cause the thread to release its resources and destroy itself when all of its processes have been completed. In the other words, when the thread is created in a joinable state, the stream socket will get stuck unless all of the threads have been destroyed. In steps 4, 5 and 6, a new detached thread is created that will call and implement the corresponding function. In step 7 when the function finishes its processes, the thread will automatically destroy itself.

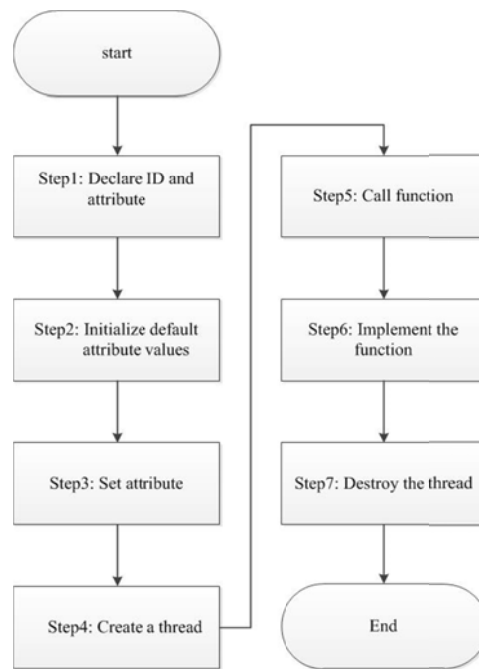


Figure 3-36 The cycle of a thread

On the contrary, the color detection function is developed in a single-threading, in this way the color detection results can be sent back to the client. Below are the specifications of the previously mentioned functions.

(1) Basic Function

The basic function within DARwIn-OP can be divided into three parts: motion, walking and speech functions. Once DARwIn-OP has received the type I packet, a portion of the motion page or line will be extracted based on the identifier and executed using its related functions. For example, if the packet is written as “A001”, it means that the robotic puppet is requested to run the motion function in accordance with the motion page (001) introduced in 3.3.1.

If the page number does not belong to the motion page, it will be translated into the units used to set the step values in the x axis for walking function use. In addition, the step values in the x axis can be manipulated in order to adjust the walking style of the robot. The range of the step values can be set from zero to thirty units. When the value is equal to zero, this represents that the robot will march in place. On the other hand, the greater value is, the longer the length of each step will be. If the step value is greater than forty units, the DARwIn-OP will begin to walk unstably. ISAP has provided the user with four walking fashions: marching in place (zero units), small steps (10 units), big steps (20 units) and strides (thirty units). Furthermore, the “Chello” packet tells DARwIn-OP to run the speech function in order to say the line hello.

(2) Kinect-based Representation

On the other hand, type II packets will provide DARwIn-OP with sufficient information to perform the Kinect-based representation. When DARwIn-OP receives type II packets, it will reserve enough memory space to temporarily store the following posture command by using the information that was received from the first packet. Next, the following postures that contain actuator's position, speed, continuous_bit, and execution time will be utilized in representation. By doing so, the key-poses captured by Kinect can be applied to the robotic puppet show.

(3) Color Detection Function

Once received the request from Node.js, the DARwIn-OP will execute the color detection function to make meaningful decisions. Hue, being one of the main properties of color, is used to implement the color detection function. CIE (International Commission on Illumination) defined Hue as an “attribute of a visual sensation according to which an area appears to be similar to one, or to proportions of two, of the perceived colors red, yellow, orange, green, blue, and purple.” [19]. Figure 3-37 shows the hue unit degree is based on a scale from 0 to 360.



Figure 3-37 Hue scale

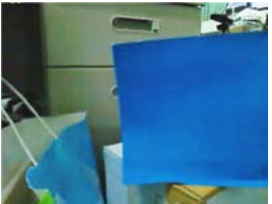
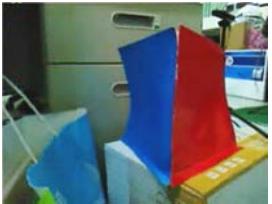

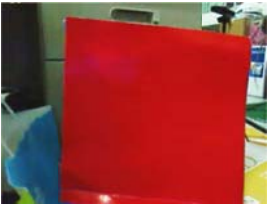
During the process of color detection, DARwIn-OP will take an image via camera, which is shown in Figure 3-38. The image is 320 pixels wide, 240 pixels tall and has a total of $320 \times 240 = 76800$ pixels. Then an array is created that contains all of the pixels in the function. Each element in the array represents a single pixel. The function uses hue degrees to calculate the similarity between each single pixel's hue and the target color under a given tolerance. If more than sixty percent of the pixels are found to be the same as the target color, the color card seen by DARwIn-OP will be able to pinpoint the pixels that are the same color as the target color.



Figure 3-38 Image via camera

Table 3-20 shows the result of color detection experiment, which is used to detect a red color that has a hue degree of zero. In addition, the valid-color bit is designed to check whether the given color is regarded as a target color or not.

Table 3-20 Result of color detection

Captured Image Color Recognition				
Amount of red pixels / total pixels	130 / 76800	10940 / 76800	25968 / 76800	46548 / 76800
Valid-color bit	False	False	False	True

There are four major colors including: Red (0 degree), yellow (55 degrees), green (139 degrees) and blue (233 degrees), which correspond with the color cards the user has presented.

3.7.2 NAO API

With the assistance of the middleware, NAOqi, the NAO is given the ability to run a variety of functions such as motion and speech by using the related modules. Since NAO is regarded as a new type actor in EPFS, it is supported by type I packets when serving as a stream socket client. The command stored in the packet will be extracted based on the identifier in order to execute a python motion file or run a speech function with a line. Moreover, both of them are executed using multi-threading in order to express the two different emotions at the same time.

3.7.3 Tablet Computer

Android is an operating system based on Linux. It is mainly designed for mobile devices such as smartphones and tablets. Android provides developers with plenty of APIs to create different applications which can be perfectly matched with tools like cameras, touch screens and sensors on smart devices. Figure 3-39 [20] shows that Android contains three major layers: operating system, middleware, and applications. The middleware can be divided in two parts, the lower part includes libraries and virtual machine; the upper is for application framework.

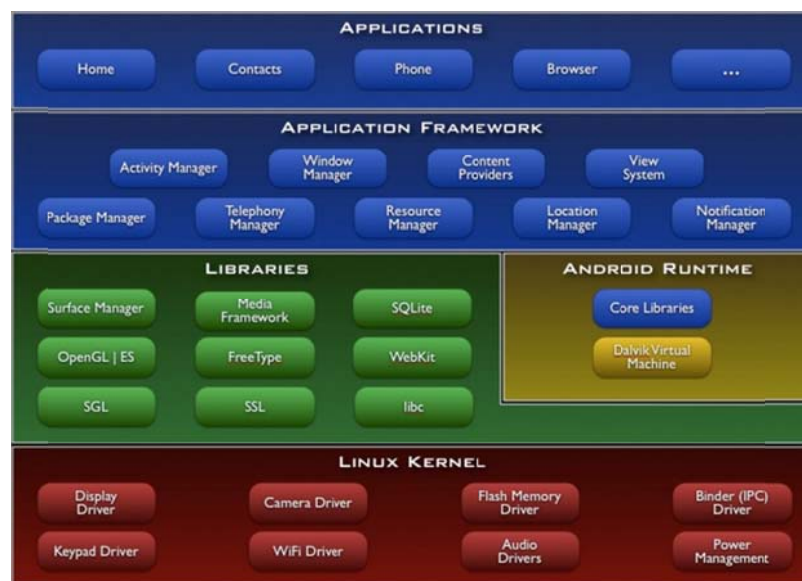


Figure 3-39 Android architecture

Unlike the normal way to program an Android application in JAVA language, the Titanium Studio IDE provides user with exclusive APIs to develop cross-platform applications by only using JavaScript, HTML, and CSS. The related Software Development Kit (SDK) should be installed at first. Titanium will pack the mobile

web application into the native application that can be run on Android, iOS or Blueberry OS.

IV. IMPLEMENTATION

This chapter describes the implementation of the platform. In addition the interface and the functions that belong to ISAP and MI are presented. Finally, the improvements made in EPFS are compared to the previously proposed platforms that were mentioned in the Related Works section.

4.1 Development Tools

(1) Sublime Text 2

Sublime Text 2 is a cross-platform text editor for coding based on Python. It's available for Linux, Windows and OS X. Sublime Text 2 supports 27 programming languages it also contains a mini map in interface and many theme options. Moreover, the main features, expandability and customization of Sublime Text allow users to have the ability to install a variety of plugins when they need to make Sublime Text more powerful. The plugins include programming language, theme, syntax highlight, etc. Therefore, Sublime Text is chosen as the main development tool for programming JavaScript, CSS, HTML and PHP languages. Below are the three steps to install plugins into Sublime Text.

- Step 1 : The manager of Sublime Text2 (packet control), should be installed during the first usage. The process of copying the Python code is shown in

Figure 4-1 [21].



Figure 4-1 The process of copying the Python code

- Step 2 : In Figure 4-2, Press Ctrl + ` to access the console and paste the Python code to install the manger and press Enter. Then restart the Sublime Text to finish installation.



Figure 4-2 Pasting the Python code into console

- Step 3 : Press Ctrl + Shift + P to open the command palette and run the related commands like install, remove or list package, as shown in Figure 4-3. Users can type and install any plugin by first executing the command “Install package”.

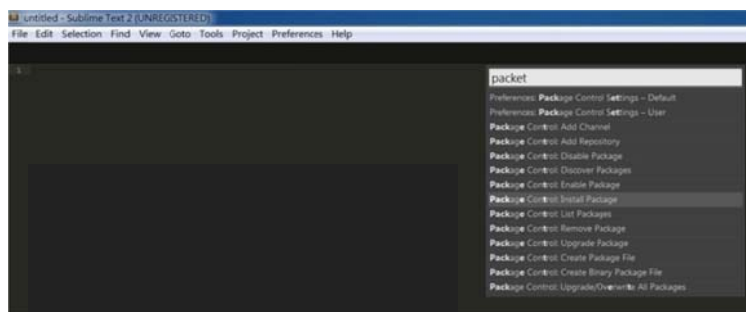


Figure 4-3 Run the related commands via packet control

(2) Titanium

In order to develop a mobile application using experience in website development, Titanium is chosen as the IDE for programming on Android. First users need to register to be a member and then log in to start using. Figure 4-4 shows that the related SDK should be downloaded and installed to work with Titanium.

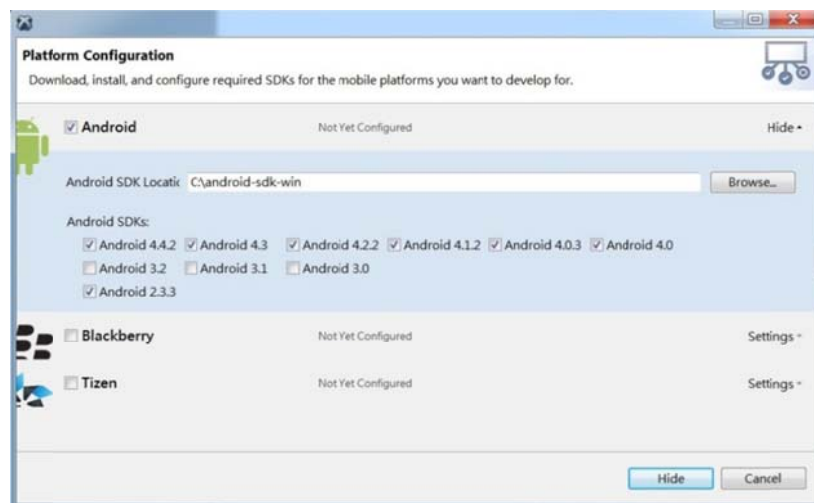


Figure 4-4 Android SDK installment

4.2 Implementation of ISAP

In order to build up a highly efficient and reliable web server, the LAMP mentioned in 2.1 is taken to simultaneously give responses to numerous requests. Furthermore, with the assistance of the hosting control panel: ISPCConfig 3, the administrator of the website is given the ability to manage multiple servers such as Apache, FTP (File Transfer Protocol), and database. The FTP server allows the administrator to update or exchange files within the website, which make the development and maintenance faster and easier.

After setting up a website environment, this thesis has utilized a relational database as depicted in Figure 4-5 that provides users with a screenplay editing service and the ability to record related data such as the screenplay details and user information.

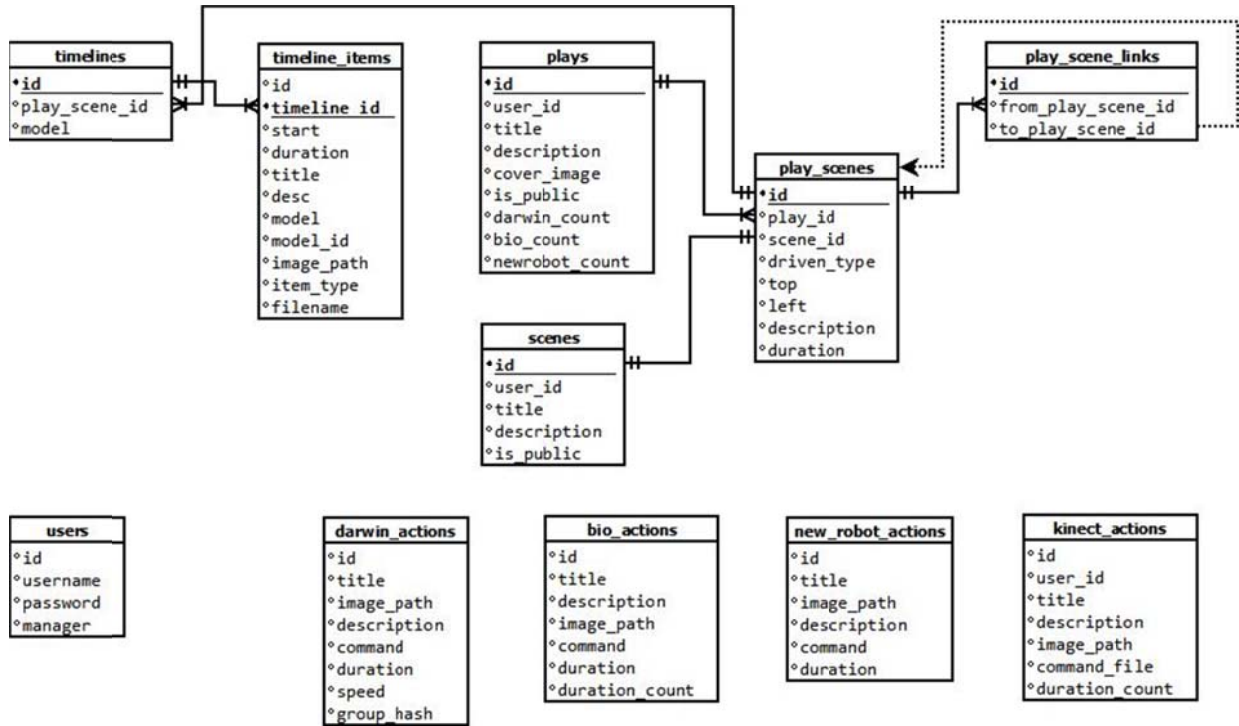


Figure 4-5 Entity relationship diagram for EPFS database

Thanks to the advantages gained by using the MVC pattern, this thesis has utilized CakePHP [22] framework to construct the architecture of the platform. In addition, the process of this website can be illustrated in Figure 4-6.

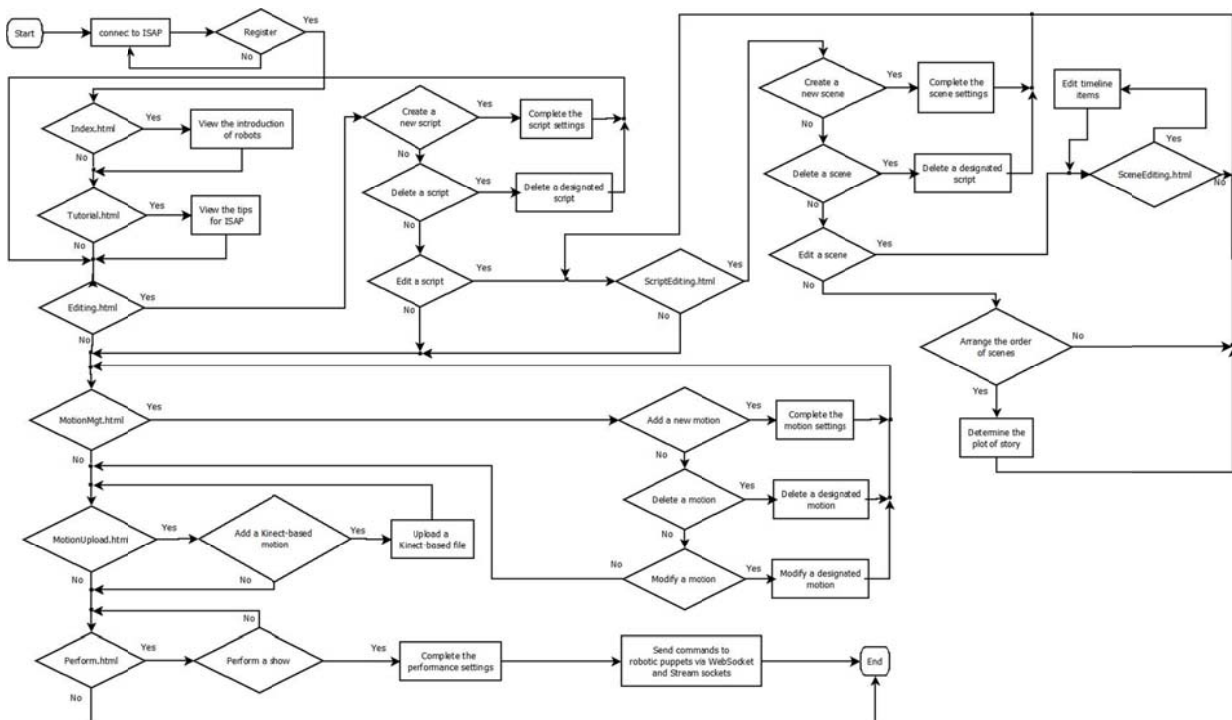


Figure 4-6 Website flow chart

4.3 User Interface of ISAP

(1) Account Registration

As shown in Figure 4-7, users need to sign up for a new account, when they login into the website for the first time. After the account information including user's name and password have been checked and accepted, the user will have the ability to access the website.



Figure 4-7 Log into website

(2) Home Page

Figure 4-8 shows when a user has logged into ISAP, the user is then led to the main page. The main page shows an introduction of the different performance robots and their specific functions as well as a newsfeed which shows the latest functions available on the website.



Figure 4-8 Home page

(3) User Tutorial

Here are some tips listed in order to help beginner users get familiar with functions of ISAP step by step, as depicted in Figure 4-9.



Figure 4-9 User tutorial

(4) Screenplay Settings

In Figure 4-10, users can edit personal screenplays. The screenplay title and description need to be filled in by the user. In addition, the user also can pick a corresponding picture as the cover of the screenplay, and then select how many robots of each type are desired.

新增劇本	
劇本名稱	外太空的朋友
劇本封面	<input type="button" value="選擇檔案"/> wally robot.jpg
敘述	來自外太空的機器人，拜訪地球的目的到底為何？究竟是敵人還是朋友呢？
機器人數量	
Darwin	1
Bioloid	1
New Robot	0
<input type="button" value="新增"/>	

Figure 4-10 Screenplay creation

Then, the newest screenplay is shown on far left side of the row, as shown in Figure 4-11. Moreover, in the new screenplay cover, the re-write and delete function are provided in the top right-hand side. In the middle left-hand side, the letters “D”, “B” and “N” represent DARwIn-OP, Bioloid and New-type robots which play roles in dramas. Once the number of robots within the screenplay has been decided, it cannot be edited without deleting the entire screenplay. The title and the description of screenplay need to be filled at the same time. Finally, user can click the “edit” button to get started designing the details of the play.



Figure 4-11 List of screenplays

(5) Scene Establishment

A whole screenplay consists of many scenes. In dramas, scripted events happen in linear sequences. On the other hand, if a user makes a meaningful choice that causes branching, the action will be called non-linear [1]. ISAP allows users to add all of the scenes that are needed in the screenplay, which include the initial, middle and last scene. In Figure 4-12, when adding a new scene, the scene's title and description need to be filled in first, and the scene selector is used to choose what automation mode and three-color card mode functions need to be utilized for the specific situation at hand. Automation mode means the scene is linearly played in an automatic fashion. In other words, the story can never be diversified. On the contrary, the color-card mode allows audiences to use a color card which can be recognized by DARwIn-OP and also can be used to pick different paths as the story progresses.

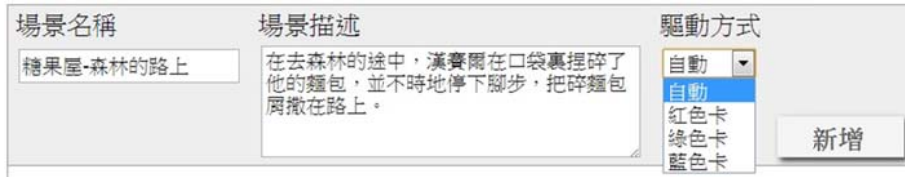


Figure 4-12 Adding a new scene

As shown in Figure 4-13, a scene box has appeared to show that the new scene has been successfully created, it has an automation mode or color-card icon on the top left side. There is a tooltip icon on the bottom of the scene box that allows the user to see a description of the scene.



Figure 4-13 Scene box functions

(6) Scene Editing

After creating a new scene, the user can double click on the scene to edit its content. The scene-editing page is divided into three major divisions: the motion library, music library, and the timeline, as shown in Figure 4-14. The motion library provides many predefined postures and motions, which are classified as either robots or Kinect motions. When dealing with DARwIn-OP and Bioloid robots, GIF images are motions and JPEG images represent postures. The name of the motion and its duration time will appear in front of the picture when mouse is placed on top of the specific motion. Secondly, the music library has a diverse collection of sound effects

and background music

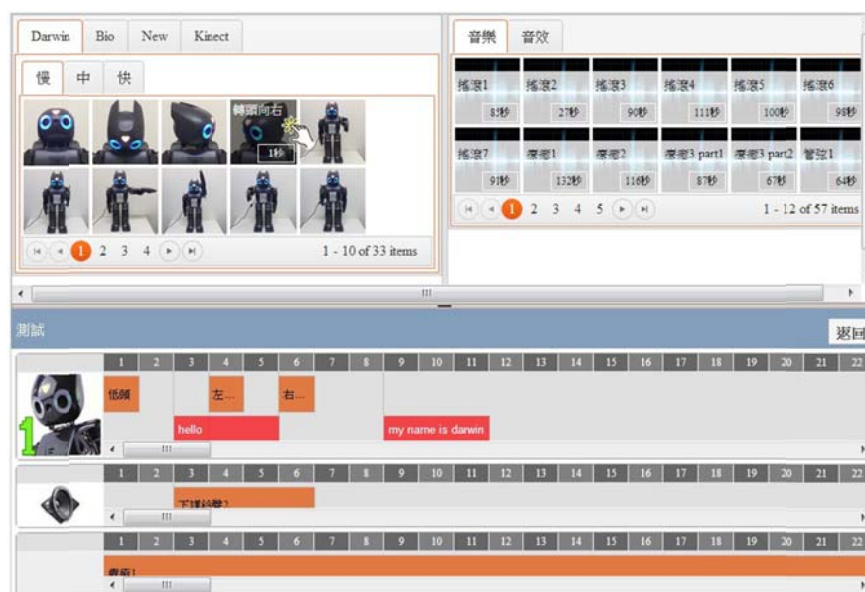
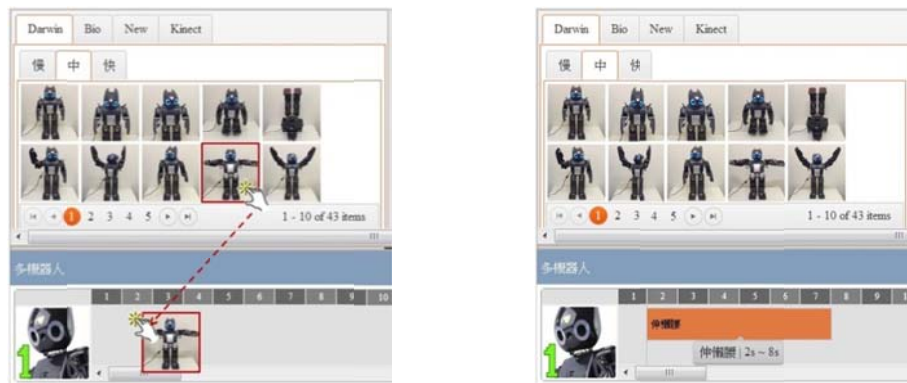


Figure 4-14 Scene-editing page

In Figure 4-14, the timeline consists of the sound effects, background music as well as the motion and speech of each robot. Each scene only has one local timeline based on seconds, which is designed to solve the problem of synchronization and allow for each performance element to be precisely arranged. The files in the motion and music library can be dragged into a specific track in order to add new event in the timeline. Figure 4-15 (a) shows the user picking up a motion file and dragging it into a track. In the next step the motion file needs to be placed at the desired spot on the timeline. Finally, the new action appears with its name on the timeline along with a box that represents the starting and ending time of the action, as depicted in Figure 4-15 (b). While taking files from the library the user must choose a file that matches the picture on the left hand side of the specific track where the action will be placed

on the timeline. For an instance, the DARwIn-OP files can only be dragged into the DARwIn-OP track on the timeline. On the other hand, it is not possible to put music files or other robots actions into the DARwIn-OP track. Only DARwIn-OP and Bioloid robots are supported by Kinect, so Kinect files cannot be dragged into any other kinds of tracks. Moreover, a user can cancel the action by double clicking on the box that represents the action on the timeline.



(a) Dragging the motion file into a track (b) New action box appears

Figure 4-15 Adding a new motion into a track

Figure 4-16 shows that robots also have a speech function. If the user double clicks on the blank area below the numbers the timeline a box will appear where the user can type what he/she wants the robot to say. If the speech box needs to be deleted, user can click the delete button on top right side.



Figure 4-16 Speech function

(7) The Relationship of Scenes

The content for a scene which the user has edited will be automatically stored in a database. After finishing the content of single scene, the linking tool is used to establish the relationships between all of the scenes in the screenplay. In Figure 4-17, the user can drag the linking tools from the source scene to the target; after this action is completed an arrow will appear between the two boxes signifying the completion of the linking process. If the user decides to delete the arrow he/she can do so by double clicking on it.

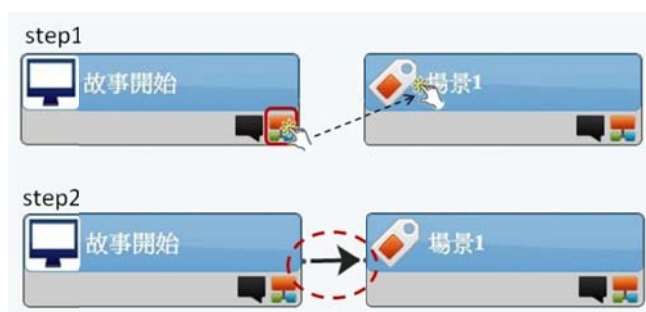


Figure 4-17 Linking tool of scene

Figure 4-18 shows a scene box that has more than two child nodes in the next level, which will cause branching at the end of the current scene. When an automatic mode box is put on the same level with other color mode boxes, the automatic mode will always be chosen and processed first.

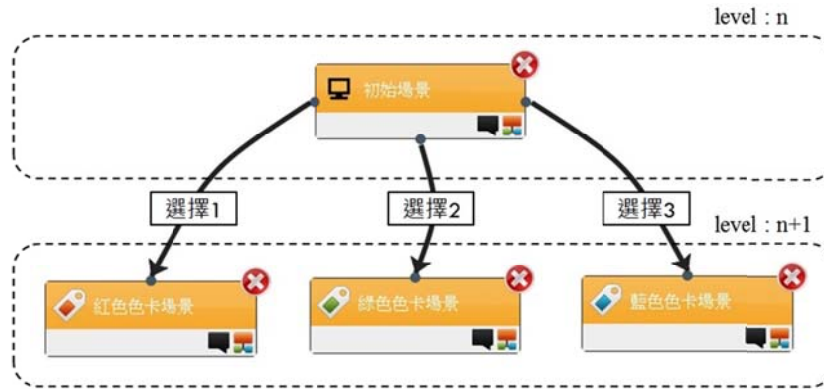


Figure 4-18 Branching

(8) Motion Management

In Figure 4-19, EPFS has provided the administrator of the website with the ability to add or modify existing DARwIn-OP and Bioloid motions as well as any type of performance robots with speech and motion functions like NAO. The GIF and JPEG are both supported as the format of images in this platform to differentiate whether this command is a motion or a posture.

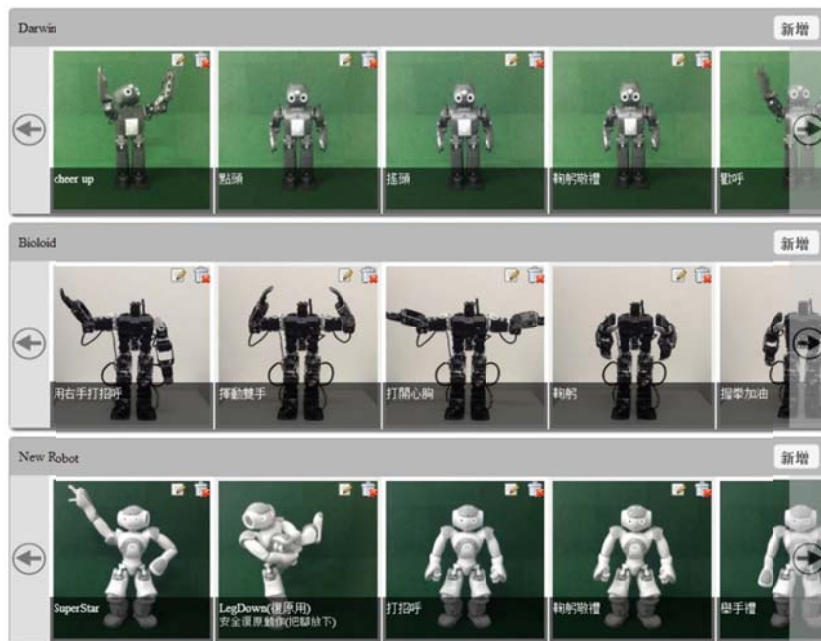


Figure 4-19 Motion management page

For instance, the administrator has the ability to add the DARwIn-OP motion by setting the motion name, corresponding image, description and motion pages in three different actuator speeds and times as shown in Figure 4-20.



Figure 4-20 Motion adding page for DARwIn-OP

Figure 4-21 shows that the Bioloid motions can be added by setting the motion name, corresponding image, description, and uploading the motion file.



Figure 4-21 Motion adding page for Bioloid

The new type robot motion can be added by finishing the different fields within the page including the motion name, corresponding image, time, command (motion file name preprogrammed within the robot), description and image addition (shown in Figure 4-22), which helps the user add the new robots motion in a clear and simple way.

The image shows a software interface window titled "新增 New Robot 動作". It contains several input fields: "動作名稱" (Action Name) with a text box; "動作圖示" (Action Icon) with a "Choose File" button and "No file chosen" text; "敘述" (Description) with a large text area; "指令" (Command) with a large text area; and "動作時間" (Action Time) with a dropdown menu. A "新增" (Add) button is located at the bottom right of the window.

Figure 4-22 Motion adding page for new type robot

(9) Kinect-based Motion Uploading for DARwIn-OP

First the user needs to install the Kinect SDK. After the imitation program is executed, the user's movement can be captured and recorded by Kinect. By uploading the imitation texture file named "kinect.txt", the user can add the motion into the robotic puppet show as shown in figure 4-23.

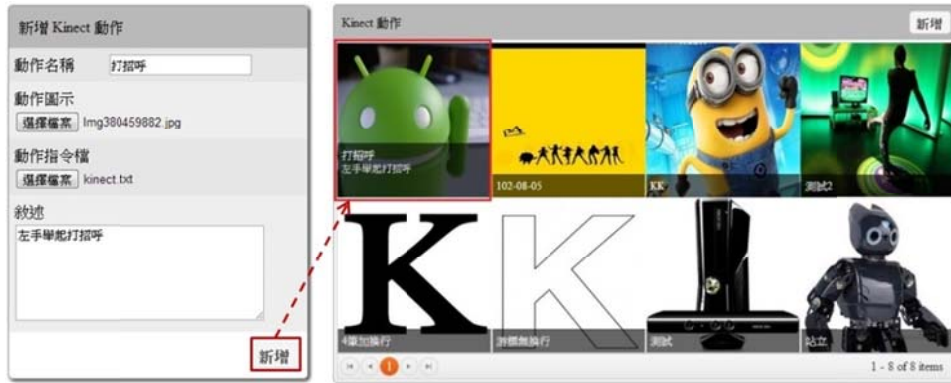


Figure 4-23 Uploading a Kinect-based motion file

(10) Performance Settings

There are three steps that need to be completed before robots can be put into motion. Figure 4-24 shows that Node.js must be opened and left running in the background. The next step is to select a screenplay and set up the IP address and port number of each performance robot as depicted in Figure 4-25. Finally, press the start button to start the screenplay.

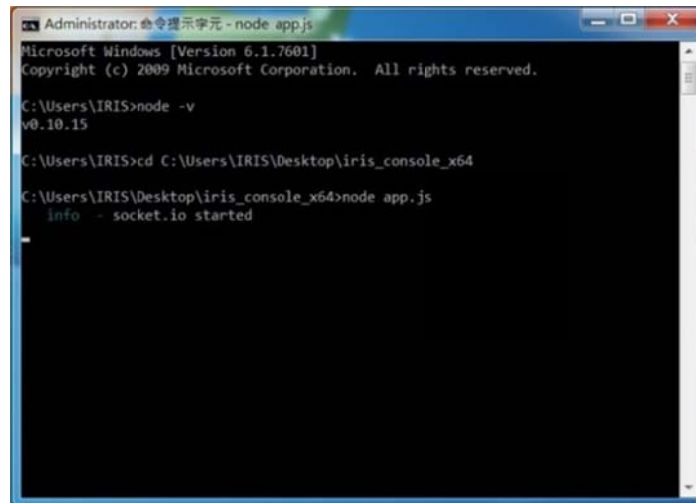


Figure 4-24 Running Node.js

The screenshot shows a mobile application interface with a grey header bar containing the text '機器人連線位址'. Below the header, there are three vertically stacked sections, each representing a different robot. The first section is titled 'Darwin 1' and contains two input fields labeled 'IP' and 'PORT'. The second section is titled 'Bio 1' and also contains 'IP' and 'PORT' input fields. The third section is titled 'NewRobot 1' and contains 'IP' and 'PORT' input fields. At the bottom right of the interface, there is a button with the text '開始播放'.

Figure 4-25 Setting IP addresses and port numbers

4.4 Android Application for Robotic Puppet Show

The application is mainly designed to make the show portable and accessible. It gives users the basic ability to perform only by touching the panel of the smart device. The tablet and performance robots should be connected a Wi-Fi network first. Once the user has logged into the application, screenplays will be automatically updated and downloaded from the ISAP website as shown in Figure 4-26. The application allows users to select a desired screenplay to perform by setting up each puppet IP and port number via stream socket. At the moment of branching, the application also has the ability to pick a different path for the story from the result of the color detection function.

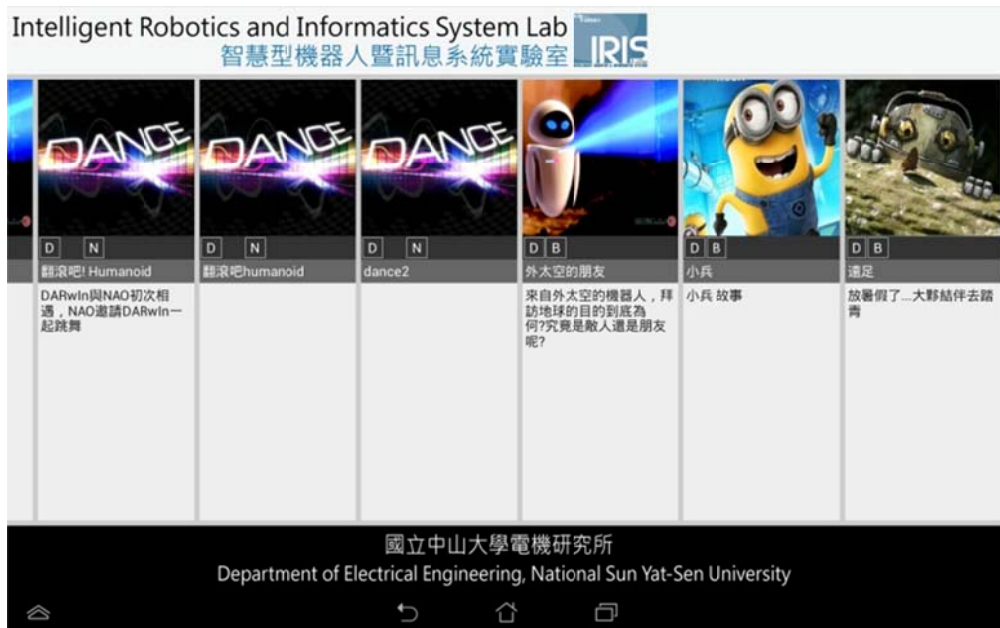


Figure 4-26 Screenplay list on tablet

4.5 Improvement

There are six essential parts of improvement in EPFS shown below and they will be compared to the proposed platform I [9]and the proposed platform II [2][3].

(1) Casting the Robotic Actors

The previous work only provides users to control the default type robots in the show. EPFS can support any kind of robot that has built-in motion and speech functions. Users can upload or modify their own robot's motion commands in this platform; those commands and lines can be arranged into timelines within a screenplay and be delivered to specific robots via sockets. In this way, users can design a dynamic show with their desired performance robots.

(2) Robotic Expression and Interaction Solution

The concept of using timelines within each scene to robotic expression and interaction between robots, sound effects and background music in EPFS is similar to how it is done by [9]. EPFS has developed visual timeline tracks within each scene which allows users to edit the details of screenplays in a more accurate and simple way.

(3) Effective Atmosphere

In stage, the entertainment elements such as lights, music and special effects are essential parts of achieving a successful performance. In order to make robotic puppets more vivid, facial icons have been presented on the panel of the smartphone to express actor's emotions [9]. EPFS utilizes sound effects and background music to enhance the robotic puppets performance for the viewer's pleasure.

(4) Dynamic Screenplays

In the proposed platforms [9][2][3], the robotic puppet show can only be performed according to the sequential arrangement within a screenplay, no matter whether the platform is event-driven or timeline based. In EPFS, the dynamic screenplay divides the story into lots of scenes, each with a flexible timeline. The order of those scenes can be easily arranged as the director desires by using GUI. Any scene node having more than two children will create branching, which means that

users have been given the opportunity to influence the development of the story. By doing so, even if the screenplay is performed many times, the plot could still be diverse.

(5) Motion Library Extension

While editing a screenplay, the authoring tool allows users to adjust the posture of robot by setting the parameters of the actuators [9]. On the other hand, the platform [2] only provides users with default postures and motions to use. Compared to these two proposal solutions, EPFS provide users with a more intuitive and simple way to add new motions based on Kinect into the platform. In addition, different Kinect based files can be shared between users. Hence, a variety of motions will make the robotic puppets more vivid and pleasing to the audience.

(6) Mobile Application

Smart devices that run on Android applications are applied to the robotic puppet show in the proposed platform [9], and EPFS. Both of the devices are responsible for sending instructions to puppets via wireless technology such as Wi-Fi or Bluetooth. The major difference is that the devices within the former platform [9] are used to receive instructions that are extracted from the screenplay. It also allows robots to implement motion, speech and facial expression functions. Thankfully, popular performance robots have built-in controllers which have turned smart devices into

screenplay interpreters and plot controllers in EPFS. Moreover, the device (tablet computer) also has the ability to synchronously play sound effects and background music. Therefore, As long as the Android devices and the robotic actors are connected to Wi-Fi, robotic puppet shows can be performed anytime. Table 4-1 shows the comparison between previous work and EPFS.

Table 4-1 Comparison between proposed platforms I, II and EPFS

Function \ Platform	proposed platform I	proposed platform II	EPFS
Type of Robots	Bioid	DARwIn-OP NAO	Any kind
Vocal Expression	Yes	Yes	Yes
Web-based Authoring	Yes	Yes	Yes
Driven Type	Timeline	Event	Timeline
Expression in Motion and Speech.	Yes	Yes	Yes
Interaction between Robots, Sound Effects and Background Music	Yes	No	Yes
Effective Atmosphere	Facial Expression	No	Sound effects Background Music
Dynamic Interaction	No	No	Yes
Motion Extension	Only Posture	No	Yes

V. CONCLUSION AND FUTURE WORK

5.1 Conclusion

This thesis provide user with a GUI to edit personal screenplays rather than a redundant and complicated way to control the robotic puppets. The screenplays created by the director can be performed with not only robotic puppets but also entertainment elements such as sound effects and background music. The addition of new robot types and the extension of motion library features make this platform more flexible and functional. The audience can interact with robotic puppets by using dynamic screenplays which gives users the opportunity to not only be a viewer but also to participate in the shows. For example, even young users who may not have the ability to edit the details of screenplays; still can make interactions with robotic puppets while branching. Thanks to the web based architecture, EPFS can be accessed by any devices with a web browser or smart devices connected to Wi-Fi for breaking the physical restrictions. As a result, the robotic puppet show can be presented any place any time and will surely entertain any audience.

5.2 Future Work

In spite of the fact that EPFS has many useful functions, there are still some parts that can be improved or modified in the future. Three possible changes are described below.

(1) Diverse Ways for Users to Interact with Robots

In EPFS, the decisions for branching are made based on the result of color detection by DARwIn-OP. Presenting different colors in front of DARwIn-OP's camera is direct and simple but it still causes some inconveniences because the user still needs to prepare additional color cards or operate the panels on the smart devices to display the colors. Therefore, in order to make the interaction more convenient, the color detection may be substituted for face or voice recognition. By doing so, more interesting and significant scenario could be created when the audience interacts with robotic puppets.

(2) More Performance Element

Sound effects and background music have been played in the show performance in order to enhance dramatic tension. In the real world, a huge screen is commonly used in concerts to project real-time videos and create an effective atmosphere.

Thus, the video can also be applied to the robotic puppet show to make the performance more attractive and complete. The additional timeline track can be added

within a scene to record the video arrangement, and the designated videos will be synchronized with the robotic puppets, sound effects and music. Next, the video will be played with a monitor or on the panel of smart device behind the robotic puppets.

(3) Key-pose Reduction

The Kinect-based motion files can be uploaded to EPFS in order to extend the robot motion library. But there are still a portion of redundant postures that are regarded as key-poses and recorded in the files. If the problem above could be solved, then the size of motion files will become smaller, due to the fact that the redundant postures will be filtered out. By doing so, the system resource and storage space can be saved.

REFERENCE

- [1] A. Rollings and E. Adams, Andrew Rollings and Ernest Adams on Game Design, New Riders, 2003.
- [2] J. J. Siao, "The Graphic Authoring Platform of Screenplays for Robotic Puppet Shows," National Sun Yat-sen University, 2012.
- [3] W. C. Wu, "The Development of Screenplay Interpreter for Multi-morphic Robots," National Sun Yat-sen University, 2012.
- [4] "The Model-View-Controller (MVC) Design Pattern for PHP," [Online]. Available: <http://www.tonymarston.net/php-mysql/model-view-controller.html>. [Accessed 03 12 2012].
- [5] "January 2014 Web Server Survey," [Online]. Available: <http://news.netcraft.com/archives/2014/01/03/january-2014-web-server-survey.html>. [Accessed 03 01 2014].
- [6] "Comparison of relational database management systems," [Online]. Available: http://en.wikipedia.org/wiki/Comparison_of_relational_database_management_systems. [Accessed 14 12 2012].
- [7] Y. A. Zheng, "Humanoid Robot Behavior Emulation and Representation," National Sun Yat-sen University, 2012.
- [8] 王森, KINECT 體感程式設計入門, 基峰資訊, 2012.
- [9] C. A. Chen, "The Authoring tool and Performance Platform for Robotic Puppets Show," National Chung Cheng University, 2011.
- [10] "Internet popularity survey," [Online]. Available: <http://www.find.org.tw/find/home.aspx?page=many&id=357>. [Accessed 03 09 2013].
- [11] C. Y. Weng, "Design and Implementation of Visual Programming Interface for Low-Cost Embedded Systems," National Chung Cheng University, 2009.
- [12] "Introducing JSON," 07 02 2013. [Online]. Available: <http://www.json.org/>.
- [13] "Node.js v0.10.24 Manual & Documentation," [Online]. Available: <http://nodejs.org/>. [Accessed 15 03 2013].
- [14] "What is WebSocket?," [Online]. Available: <http://www.websocket.org/>. [Accessed 18 03 2013].
- [15] "Stream socket," [Online]. Available: http://en.wikipedia.org/wiki/Stream_socket.

- [Accessed 10 03 2013].
- [16] "User's Manual-Dynamixel AX-12," [Online]. Available: [http://www.electronickits.com/robot/BioloidAX-12\(english\).pdf](http://www.electronickits.com/robot/BioloidAX-12(english).pdf). [Accessed 23 02 2013].
- [17] I. Ha, Y. Tamura, H. Asama, J. Han and D.W. Hong, "Development of open humanoid platform DARwIn-OP," The University of Tokyo, Virginia Tech.
- [18] A. Robotics, "Hardware Platform of NAO," [Online]. Available: <http://www.aldebaran-robotics.com/en/Discover-NAO/Key-Features/hardware-platform.html>. [Accessed 26 07 2013].
- [19] "Hue," [Online]. Available: <http://en.wikipedia.org/wiki/Hue>. [Accessed 17 04 2013].
- [20] "Android Architecture," [Online]. Available: http://elinux.org/Android_Architecture. [Accessed 19 10 2013].
- [21] "Package Control," [Online]. Available: <https://sublime.wbond.net/installation>. [Accessed 18 12 2012].
- [22] "Cookbook 2.x," [Online]. Available: <http://book.cakephp.org/2.0/en/index.html>. [Accessed 30 11 2012].