**Robot body language telepresence PROJECT CHARTER**

Version: v 1.0

Date: 20 Mar 2019

Sponsor: Virtual Experience Laboratory

Group Number: V10

Authors: Minh Doan Quang, Lachlan Clulow, Jesse Buhagiar,

Sreeja Manikyam

***Commercial - in – Confidence***

**Document Control**

**File Directory** Documents/Project\_Charter.docx

**Distribution Version Issued Recipient Entity / Position** V 1. 24 Mar 2019 Vic Ciesielski Supervisor

**Amendment History Section Page Version Comment** 1 - 8 2- 12 1.0 Initial Project Charter draft write up

**Staff or Entities Consulted Name Position / Organization** Ian Peake Virtual Experience Laboratory

Add rows as needed. If not relevant, enter N/A.

**Related Documents Name Author Description**

H:\DATA 2006\Templates\Template - Project Charter.doc - 2 -

***Preface***

The purpose of this document is to outline the Charter for Robot Body Language Telepresence Project. It serves as an agreement between the project team, the sponsor and the supervisor. It outlines the project’s purpose and how the project will be approached, resourced, managed and delivered. Any amendments after this document has been signed off will be via addenda.

H:\DATA 2006\Templates\Template - Project Charter.doc - 3 -

**Table of Contents**

**1 PROJECT SUMMARY** 5

**2 PROJECT SPONSOR** 6

**3 STAKEHOLDERS AND END USERS** 7

**4 APPOINTMENT OF PROJECT LEADER** 8

**5 PROJECT TEAM MEMBERS** 9

**6 PROJECT METHODOLOGY AND APPROACH** 10

**7 PROJECT GOVERNANCE** 11

**8 PROJECT SCOPE & DELIVERABLES** 12

H:\DATA 2006\Templates\Template - Project Charter.doc - 4 -

**1 Project Summary**

Our Project’s mission is to research and develop a telepresence application for the robot ‘Rosie’ at Virtual Experience Laboratory. The intent of the project is to allow Rosie to mimic human body movement using data from a Microsoft Kinect device. Rosie is a Baxter robot manufactured by Rethink Robotics that is longer available on the market. Therefore, Rosie is considered one of the legacy robots that RMIT has at the moment. This project will allow RMIT to extend the useful lifetime of Rosie, as the RMIT Vietnam Campus also have another Baxter robot. This project will allow the Baxter robots from both campuses to behave as human avatars. They will help convey the body language of the speakers from one side of the telepresence call to the other. Rosie will be able to retrieve data from a human skeleton sensor; in this case it will be the Microsoft Kinect. Rosie will be able to recreate the movement of a human by using the data retrieved from the sensor.

Objectives:

● Tracking human skeleton using kinect SDK provide by Microsoft

● Communicating between Sensor and Rosie, sending data and retrieve data

● Handing retrieved data and turn it into actual robot movement

● Improve the legacy code from previous project group

H:\DATA 2006\Templates\Template - Project Charter.doc - 5 -

**2 Project Sponsor**

The project sponsor is Professor Heinz Schmidt of Virtual Experience Lab. Virtual Experiences Laboratory is a sub part of RMIT in which create a space for students and industry to have experiences on advanced technology. Providing a range of industry application including manufacturing, electronic engineering, information technology, analytics, games and medical imaging. Also, VXLab act as the middleman to connect industry and student together. Allow students to have access to real industrial project and also allow industry to found potential candidate.

Focus Technology:

● Virtual Reality

● Cloud Computing

● Robotics

Office located at: RMIT building 91 Victoria, Carlton VIC 305

H:\DATA 2006\Templates\Template - Project Charter.doc - 6 -

**3 Stakeholders and End Users**

NAME & ORGANIZATION

PROJECT ROLE PROJECT RESPONSIBILITIES

Professor Heinz Schmidt Virtual Experience Laboratory, Executive

N/A Project Sponsor

Ian Peake Virtual Experience Laboratory

Product Owner / Lab Manager

Mentoring and supervising development team

Mentoring and supervising development team

Vic Ciesielski, RMIT

Supervisor Supervising and monitoring the

project

H:\DATA 2006\Templates\Template - Project Charter.doc - 7 -

**4 Appointment of Project Leader**

The project leader is Lachlan Clulow. We all as team decided that Lachlan would be leading us during the second meeting with the supervisor. He is proficient in C, Java and python programming skills. He is very familiar with Linux command line and has very good software engineering project management skills

H:\DATA 2006\Templates\Template - Project Charter.doc - 8 -

**5 Project Team Members**

The project team members and their respective roles are:

Print Name: Mr. Minh Doan Quang

Student Number: S3608452

Role: Developer / Scrum Member

Print Name: Mr. Lachlan Clulow

Student Number: S3682356

Role: Developer / Scrum Master

Print Name: Mr. Jesse Buhagiar

Student Number: S3600244

Role: Developer / Maintainer / Scrum Member

Print Name: Mr. Sreeja Manikyam

Student Number: S3705865

Role: Documentation / Tester

H:\DATA 2006\Templates\Template - Project Charter.doc - 9 -

**6 Project Methodology and Approach**

The approach that our project team has selected for delivering the project is “Scrum”. It is an Agile approach planning and prioritising backload which involves in higher tasks selection for development. Scrum is highly sensitive to customer needs with excellent communication and transparency. It is highly flexible to change. Acceptance criteria defines the exact behaviour of the product. The key concepts we are following in this approach are:

● The product backlog

● Sprint

● Sprint review

● Sprint retrospective

H:\DATA 2006\Templates\Template - Project Charter.doc - 10 -

**7 Project Governance**

The Governance model is as follows:

The project governance hierarchy is flat, with all members getting an equal say on input with regards to project direction and code. Any indecision and conflict of ideas will be settled via team vote (though with how large the scope of the project is already; this shouldn’t be too much of an issue). We aim to meet up at minimum weekly to discuss and test any new features that have been implemented via the virtual ROS session available on Linux, as well as to get some time in with the actual robot hardware. Everyone in the group will contribute to the ongoing programming efforts of getting the system up and running (i.e, role of “Developer” in Section 5 above), however there are individual roles. Minh is in charge of solutions for robot arms movement and human skeleton drawing algorithm , Lachlan’s main responsibility is project leader and providing development assistance, Sreeja’s primary role is handling documentation and application testing, and Jesse is in charge of maintaining the git repository, as well as the docker file.

Scope creep will mainly be settled by discussing with Ian. As there is a fair amount of different modules and subsystems to play with, it’s very possible to go crazy with implementing features, however, by discussing what is actually required with Ian, we’ll be able to keep the project nice and on track with what’s required with regards to “deliverables”. Any milestones achieved will be signed off and demonstrated to him before continuing with development on the project.

H:\DATA 2006\Templates\Template - Project Charter.doc - 11 -

**8 Project Scope & Deliverables**

● Scope ○ Determine and establish version control, resources, required packages, and

materials

○ Determine project roles

○ Source project resources

○ Research ROS environment

○ Research Kinect integration in ROS environment

○ Research Rosie (Baxter + Mobility Base) integration in ROS environment

○ Research Docker containers

○ Research cob body tracker project integration

○ Integrate Kinect sensors and cob body tracking in ROS environment

○

● Deliverables

○ Sign off on safety briefings with Lab Manager (end of week 3)

○ Sign off first project charter with project supervisor (start of week 4)

○ Set up ROS environment and apply motion commands to Rosie (end of week 4)

○ Apply correct calibration of Kinect Sensors for use in ROS (end of week 4)

○ Submit to Lab Manager a Docker container to support ROS environment (end of week 5)

○ Lab manager approval of basic motion commands to Rosie (end of week 5)

○ Lab manager approval of Kinect Sensor data collection (end of week 5)

○ Lab manager approval of human to robot joint motion conversion algorithm/software integration (end of week 6)

○ ○ ○ Submit Project for Supervisor Approval (end of week 12)

1. Converting Human joint movement to robot joint movement a. Research methods to make this conversion

i. Inverse Kinematics 2. Kinect calibration

a. Research and apply correct calibration of kinect devices to properly apply

kinect skeletal tracking 3. ROS development environment

a. Research ROS development environment b. Analyse previous group project’s implementation of ROS and expand upon

this c. Develop and test skeletal tracking d. Apply human joint movement to robot joint movement e. Apply movements to rosie 4. Docker file

a. Research docker file development b. Test and develop docker file to set up Telepresence project environment 5. Cob body tracker project

a. Research cob body tracker project b. test and apply cob body tracker in rvis

H:\DATA 2006\Templates\Template - Project Charter.doc - 12 -

**ISSUES REGISTER**

**Project : Robot Body Language Telepresence Project No: <Enter Project Number> As At: <Enter Date Updated>**

**No Description Impact Date**

**Raised**

**Date Closed**

*0 Sample – Delete this row*

*after Register has been created*

**Phase Raised Allocated to Resolution**

**Strategies / Actions** *Describe the strategies or actions that will be required to close / resolve the issue.*

*UPDATE: as actions progress and are closed..* 1 Virtual Machine was too

slow to run Simulation - Baxter SDK

*C – ritical*

*Planning*

*Name/s of H – igh*

*Analysis*

*person/s resolving M – edium*

*Development*

*the issue L - ow*

*Testing Implementation Post Imp.*

Planning Minh Doan Quang Install Dual Boot

Ubuntu instead of Virtual Machine 2 Gazebo Simulation fail to

run on Ubuntu 18.0.0

Medium 16/03/2

019

Planning Minh Doan Quang Reinstall Ubuntu

16.0.0 3 Kinect disable by itself

some time

Medium 19/03/2

019

Development Minh Doan Quang Rerun kinrosie

docker container with people\_detection package in rosie/vxlab 4 Code after Refactor with new library unable to run

Critical 01/04/2

019

Development Minh Doan Quang Debug

5 Unable to implement

ROSBridge

Medium 15/04/2

019

Development Lachlan Clulow Help Minh’s to

implement ROSBridge 6 Unable to connect to

WiFi/download ros packages on Ubuntu 14.04

High 29/04/2

019

Planning Sreeja Manikyam Reinstall Ubuntu

by dual boot but with 16.0.0 version

7 TroubIe in installing on to Medium 21/03/2 Development Sreeja Manikyam Memory extension 1

Medium 17/03/2

019

docker containers to run Gazebo simulation

019 from windows and

merging it with Ubuntu 8 Kinect sensor calibration

too difficult to achieve

Medium 18/4/20

19

Development Lachlan Clulow Use default

freenect2 driver calibration settings 910 ROS Kinetic packages fail

to launch on Arch Linux due to missing drivers

Critical 03/4/20

19

Planning Jesse Buhagiar Reformat Disk and

install a compatible Operating System (Ubuntu 16.04) 11 ROSLink fails to map

custom topic/subscriber

High 15/5/20

19

Development Lachlan Clulow &

Jesse Buhagiar

Use existing topic/subscriber to circumvent.

Impact = Impact on the Business or project in the event the issue is not resolved within the required time.

2

**RISK REGISTER**

**Project : VXLab Robot Body Language/Telepresence Project No: V10 As At: 2/6/2019**

**No Description Impact L’hood Mitigation Strategies Contingency Plan** *0 Sample – Delete this row after*

*Register has been created*

*H – igh M – edium L - ow*

*H – igh M – edium L - ow*

*Describe the strategies to mitigate the risk, ie, preventative measures and monitoring mechanism*

*Describe the contingency plan in place, to be implemented in the event that the risk is realized.*

1 Unable to implement port

forwarding for network solution to connect two baxters with kinect as port forwarding is dependent on dynamic ip address

H M Developing shell script to

update dynamic IP addresses of all components from time to tim.

Use rosbridge websocket server as a proof of concept instead of a scalable and reliable system

2 Unable to correctly calibrate the

Kinect sensor

M H Use default calibration settings

3 Unacceptable Latency incurred

by introduction of ROSLink Relays

H M Ensure the networking

solution, servers are optimally resourced and located for fast transfer of data over the roslink protocol

Look at alternative server hosting options, different server locations and hardware/software configurations

456789101H:\Data\Latest 23 Feb 2005\Lateset 20 Feb 2005\Templates\Your Software Templates\70 Mgt & control\Template - Risk Register.doc

Impact = Impact on the Business or project in the event the risk is realized L’hood = Likelihood of risk being realized

2H:\Data\Latest 23 Feb 2005\Lateset 20 Feb 2005\Templates\Your Software Templates\70 Mgt & control\Template - Risk Register.doc

**Project Robot Body Language Telepresence Development Guide**

Version: V1.0 Date: 22-5-2019 Sponsor: Virtual Experience Laboratory Author: Jesse Buhagiar

***Commercial - in – Confidence***

H:\DATA 2006\Templates\Template - Project Charter.doc - 2 -

**Document Control**

**File Directory** V10.pdf

**Distribution Version Issued Recipient Entity / Position**

V 1.0 2/6/2019 Vic Ciesielski Supervisor

**Amendment History Section Page Version Comment** 1 - 6 6 - 11 1.0 Development guide write up

Add a row for each section update or consolidate if changes are minimal. NOTE: Changes should be tracked within the document if the document is to be re-distributed, so that the audience can quickly see the changes.

**Staff or Entities Consulted Name Position / Organization** Ian Peake Virtual Experience Laboratory Add rows as needed. If not relevant, enter N/A.

**Related Documents Name Author Description** <Enter Document Name> <Enter

Author>

<Enter Document Description

Add rows as needed. If not relevant enter N/A.

H:\DATA 2006\Templates\Template - Project Charter.doc - 3 -

***Preface***

The purpose of this document is to outline a technical guide line for new developers or anyone who is going to join the team and start development.

H:\DATA 2006\Templates\Template - Project Charter.doc - 4 -

***Table* of Contents**

1Project Summary.................................................................................................................... 6 2Setup client/developer machine..............................................................................................6 3Code/Comment Standard.......................................................................................................6 4Version Control Policy.............................................................................................................7 5Deployment ............................................................................................................................ 7 6Getting Started ....................................................................................................................... 8

H:\DATA 2006\Templates\Template - Project Charter.doc - 5 -

**1 Project Summary** The *Robot Body Language Telepresence* is an attempt at using an existing humanoid, anthropomorphic robotics platform called Baxter, by the now defunct “Rethink Robotics”. The sponsor of this project, Ian Peake of the RMIT VXLAB, is aiming to get better use out of the Baxter platform by allowing the two that RMIT own (one in Melbourne, Australia and the other in Saigon, Vietnam) to “communicate” with each other by having the movements captured from a Kinect camera in Melbourne to be transmitted and translated to real movement by the robot in Vietnam. It is also something cool the University can show off at Open Day. It is our objective to:

1. Turn skeletal information send to the robot (in (x, y, z) format) into robot movement 2. To allow networking of two robots and to send information between them over the

internet.

**2 Setup client/developer machine** The system is somewhat complex to set up, with a lot of software dependencies and specific versions required for a developer to get up and running. Currently, the software required is as follows:

1. Ubuntu Wily or Xenial, either i386 or amd64. Debian Jessie amd64 is also supported. 2. ROS Kinetic 3. Software repositories found at: https://github.com/rmit-s3608452-Doan-

QuangMinh/Robot-body-language-telepresence---VXLAB https://github.com/imchockers/baxter-sim-demo https://github.com/imchockers/kinect\_based\_arm\_tracking

For the purpose of this guide, we’ll be using Ubuntu. To install ROS Kinetic on Ubuntu, follow the tutorials on the following page: http://wiki.ros.org/kinetic/Installation/Ubuntu

A Docker image can be used for the client’s machine, however this is currently unsupported.

**3 Code/Comment Standard**

To keep the code clean, there has been strict adherence to the *Python Coding Convention* (aka *PEP 8*), which is the format created for the Python Programming language. It is important that a developer follow this style. Python, being an interpreted rather than compiled language, will throw an interpreter fault if there is a discrepancy with formatting (mostly related to indentation). The key rules of this standard are as follows:

1. There are 4 *spaces* per indentation. Tabs are not allowed. 2. An indent should be inserted at the beginning of each new line where there is a block

of code, e.g a function or an if/else block 3. Line lengths should not exceed 79 characters The full style guide can be found here: https://www.python.org/dev/peps/pep-0008

The commenting style follows the Doxygen commenting standard, which includes macros and keywords to generate our own documentation in HTML form that can be uploaded to a web server. A reference to this is found here: http://www.doxygen.nl/manual/docblocks.html#pythonblocks

An example of both the commenting style and coding style is shown below

'''

Vector Length

H:\DATA 2006\Templates\Template - Project Charter.doc - 6 -

Returns the length of a given vector, <x, y, z>

@param v The vector we want to get the length of

@return The Length of the vector, @ref v ''' **def** vec3\_length**(**v**):**

**return** math**.**sqrt**((**v**[**0**] \*** v**[**0**]) + (**v**[**1**] \*** v**[**1**]) + (**v**[**2**] \*** v**[**2**]))**

**4 Version Control Policy**

Version control follows a custom version *Gitflow* style of commiting. Regular *Gitflow* dictates that the master branch is only for version releases and that a separate development branch should be set up. However, as we have a relatively small team, we have circumvented this, and all commits are pushed to master. All work MUST be done on a separate fork; pushing directly to the source repository is highly frowned upon. An example of someone adding a feature is described below:

1. The developer types *git pull -u upstream master* (assuming upstream is the name of

the remote of the upstream repository) to update their master 2. The developer then types *git checkout -b my\_branch* to create a new branch called

*my\_branch* and then check it out. 3. Features are then developed on this branch 4. When the developer is ready, the branch is pushed back to their *local* forked

repository on GitHub, and a pull-request is opened.

Once features have been reviewed (using GitHub’s pull request review section), the feature is pulled into the upstream (master) repository’s master branch, and the cycle may begin again.

**5 Deployment**

● Download ROS as described in section 1.

● For real time Kinect 2.0 tracking, install freenect2 drivers via the steps detailed: https://github.com/code-iai/iai\_kinect2#install

● Install rosbridge: http://wiki.ros.org/rosbridge\_suite

● Install this project packages: https://github.com/imchockers/kinect\_based\_arm\_tracking.git

● If you would like to connect two Baxter Robots via ROSBridge, then run the HTML client/server file. This will open a ROSBridge websocket that will allow a full-duplex Client/Server interaction. This is detailed in the readme files within the git project.

● For project simulation, install Docker : https://www.docker.com/

● Install Simulation environment and build using Docker: https://github.com/imchockers/baxter-sim-demo

H:\DATA 2006\Templates\Template - Project Charter.doc - 7 -

**6 Getting Started**

<Illustrate and explain one function, how to start, coding on front end, back end, setting and deployment, screen shot on running example>

For simulated baxter environment docker container:

1. Clone the baxter sim git repo:

a. git clone https://github.com/imchockers/baxter-sim-demo.git b. cd baxter-sim-demo 2. Build the docker container:

a. ./build b. This may take some time depending on your machines configuration,

particularly, this step:

4. Run the docker containers using provided run file:

a. ./run b. May need sudo permissions 5. To enter docker container bash shell do:

a. docker exec -it vxlab-baxter2 bash b. Should be presented with shell like:

H:\DATA 2006\Templates\Template - Project Charter.doc - 8 -

3. You should see this upon build:

H:\DATA 2006\Templates\Template - Project Charter.doc - 9 -

6. To begin Baxter simulator do:

a. ./simstart b. open a browser and go to:

b.i. http://localhost:8080/vnc\_auto.html c. Wait for baxter to appear in the simulator and the following log message to

appear in the simulator terminal:

c. Open ottserver.html in a browser and check the web console to ensure

connection has occurred

H:\DATA 2006\Templates\Template - Project Charter.doc - 10 -

7. (Optional) For local testing, do:

a. ./local\_arm\_track\_sim b. Wait for baxter to raise his arms into a T-pose in the simulator window c. Play a /tf bagfile such as the following (should cause baxter to do something

resembling a wave):

c.i. source rosenv.sh c.ii. rosbag play 2019-03-27-14-20-32.bag /tf:=/tf\_old

8. For networked operation do:

a. ./pose\_arm\_track

Get ottserver.html from: https://github.com/imchockers/kinect\_based\_arm\_tracking.git b. Edit and save ottserver.html as follows:

b.i. where var ros1 and ros2 are defined, change the url field the

hostnames or ip addresses of the two machines to be connected as below (post 9090 is the default port that rosbridge uses):

d. If playing back rosbag files do:

d.i. rosrun time\_jump time\_jump.py d.ii. rosbag play 2019-03-27-14-20-32.bag /tf:=/tf\_old d.iii. See Rosie or baxter simulator on another machine move in response

to the bagfile being played.

H:\DATA 2006\Templates\Template - Project Charter.doc - 11 -

**Test Management Document for Robot Body Language Telepresence**

Version: V1.1 Date: 1 June 2019 Sponsor: Virtual Experience Laboratory Author: Sreeja Manikyam, Lachlan Clulow

PROJECT PROPOSAL – TEST MANAGEMENT DOCUMENT, ROBOT BODY LANGUAGE TELEPRESENCE

PROJECT PROPOSAL – TEST MANAGEMENT DOCUMENT, ROBOT BODY LANGUAGE TELEPRESENCE

**Document Control**

**Distribution Version Issued Recipient Position** V <Enter No.> <Enter Date> <Enter Full Name> <Enter Position> V 1.1 3/6/19 Ian Peake Lab

Manager/Supervisor

**Amendment History Section Page Version Comment** 1 - 8 4-7 1.1 Test Schedule final write up compilation

.**Staff or Entities Consulted NAME Position / Organization** Ian Peake Virtual Experiences Laboratory

**Related Documents Name Author Description** <Enter Document Name> <Enter

Author>

<Enter Document Description>

2

PROJECT PROPOSAL – TEST MANAGEMENT DOCUMENT, ROBOT BODY LANGUAGE TELEPRESENCE

**Table of Contents**

1Introduction............................................................................................................................. 4 2Objectives and Tasks .................................................................................................................................................. 4 2.1Objectives.......................................................................................................................4 3Approach Overview................................................................................................................. 4 4Scope...................................................................................................................................... 5 5Assumptions...........................................................................................................................5 6Functional Requirements........................................................................................................5 7Non Functional Requirements.................................................................................................5 8Test Schedule ........................................................................................................................ 6

PROJECT PROPOSAL – TEST MANAGEMENT DOCUMENT, ROBOT BODY LANGUAGE TELEPRESENCE

**1 Introduction**

<The purpose of this document is to list out the main objectives, assumptions, requirements, schedules and brief explanation of tests that the team has conducted. The tests were basically part of implementation process.>

This is the test management document for the Robot Body Language/Telepresence Project. The outcome of the project is to develop a system whereby a user or users can control a robot (Rosie/Baxter) using their body movements. The body language/motion is captured by a depth sensor (Microsoft Kinect 2.0) and applied to the robot to mimic the actions of the user. This user and robot should be able to be networked such that the user and robot can be in two separate networked locations and the robot behaves as the user’s avatar.

**2 Objectives and Tasks**

**2.1 OBJECTIVES**

Unit testing of each of the components in the project to ensure that each component functions:

● Kinect sensor data output

● cob\_people\_perception ingest and publishing body tracking data

● Conversion of cob\* data to joint angles

● Publishing joint angle data

● Network transfer of joint angle data

● Client subscription to networked joint angle data

● Application of joint data to Rosie

Integration testing of each component to ensure the flow of data from kinect sensor to robot movement.

**3 Approach Overview**

● Client side publishing:

○ Kinect sensor data output

■ Ensure kinect sensor is running,

■ Visual inspection, is it powered on?

■ Data flow inspection are /kinect2/\* topics being published?

● e.g. rostopic list | grep kinect2

● e.g. rostopic echo /kinect2/sd/image\_ir/compressed

○ cob\_people\_perception ingest and publishing body tracking data

■ Ensure cob\_people\_perception package is running

● rostopic list | grep body\_tracking

■ Test user tracking data output

● rostopic echo /tf | grep cob

● Have a user stand in front of the kinect camera and see if there is output

○ Conversion of cob\* data to joint angles

■ rostopic list | grep socket

○ Publishing joint angle data

■ rostopic echo /right\_e0\_socket

● Server:

○ Ensure server connects to clients

■ Open ottserver.html in browser and check console log

○ Network transfer of joint angle data

■ On subscribing client side:

4

PROJECT PROPOSAL – TEST MANAGEMENT DOCUMENT, ROBOT BODY LANGUAGE TELEPRESENCE

● rostopic list | grep /right\_e0

● Client side subscribing:

○ Client subscription to networked joint angle data

■ rostopic echo /right\_e0

○ Application of joint data to Rosie

■ Visual inspection of Rosie, or baxter simulator responding to data input

**4 Scope**

Testing mainly applies to the flow of data within the ROS system and the successful network traversal of this data, all modules in the flow of data will be tested to ensure that the data reliably passes from one module to the next as specified in the architecture document.

**5 Assumptions**

All modules are validated to run in an Ubuntu 14.04 environment running a compatible version of ROS (kinetic or indigo).

All testing should be consistent whether running with Baxter robot units or simulated baxter units.

**6 Functional Requirements**

● System must capture image data and output point cloud data

● System must input point cloud data and convert it to a stream of user joint data

● System must be able to track one specific user at a given time

● System must be able to convert joint data to angles between specific joints for a specified user

● System must be able to transfer joint angle data between ros environments over a network

● System must be able to apply joint angle data to a robot avatar

**7 Non Functional Requirements**

● System shall not incur significant latency due to processing and network transfer of data

● System shall be reliable such that it must remain running for long periods of time without the need to reset the system

● System shall work in a variety of configurations, baxter-to-baxter, baxter-to-simulator, simulator-to-simulator etc.

PROJECT PROPOSAL – TEST MANAGEMENT DOCUMENT, ROBOT BODY LANGUAGE TELEPRESENCE

**8 Test Schedule**

Refer to Test Schedule document

6

PROJECT PROPOSAL – TEST MANAGEMENT DOCUMENT, ROBOT BODY LANGUAGE TELEPRESENCE

**Test Schedule**

***Task Name Date Related Scripts Comments Expected Result Test result*** Kinect Sensor Testing 15/3/19 rostopic echo /kinect2/image\_ir/compressed Kinect

configured, data is output successfully

A stream of ir image data appearing in the console

Success

cob\_people\_perception 15/3/19 rostopic echo /tf | grep cob Cob body

tracking data is output, cob contained in the child frame id of relevant /tf transform frames

A stream of user joint locations in 3d space

Success

User tracking 27/3/19 rostopic echo /user\_tracking

Kinect\_based\_arm\_tracking tf\_user\_tracker.py

Should output a specific user id when there is at least one user in the frame, after 10 second of not being able to track a user it should either switch to another user or state tracking None

Success

User joint data conversion

4/4/19 rostopic echo /right\_e0\_socket

Kinect\_based\_arm\_tracking tf\_listen\_socket\_pub.py

Should output 64 bit float values on a topic for each robot arm joint

Success

Apply joint angles data to rosie

17/4/19 Kinect\_based\_arm\_tracking

tf\_listen\_v8\_controller.py

Rosie should respond to joint angles data with motion of arms

Success

Networking 29/5/19 Kinect\_based\_arm\_tracking

ottserver.html

Open ottserver.html in a browser and edit the ros1 and ros2 host addresses to the addresses of the

Joint angle data publishing on /\*\_socket topics in one client should be republished in the destination client node

Success

PROJECT PROPOSAL – TEST MANAGEMENT DOCUMENT, ROBOT BODY LANGUAGE TELEPRESENCE

rosbridge websocket servers

**Technical Solution Design for Robot Body Language Telepresence**

Version: V1.0 Date: 1 May 2019 Sponsor: Virtual Experience Laboratory Author: Sreeja Manikyam,

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

***Commercial - in – Confidence***

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

**Document Control**

**Distribution Version Issued Recipient Position** V 1.0 2/6/2019 Vic Ciesielski Supervisor

**Amendment History Section Page Version Comment** <Enter Doc. Section No.>

<Enter Page No.>

<Enter Version No.>

<Enter Comments to explain the reason for the document text or other changes,

e.g., Updated text after walkthrough with the stakeholders, or

e.g., Updated section after technical consultation>

. **Staff or Entities Consulted NAME Position / Organization** <Enter Name> <Enter Position or Organization>

**Related Documents Name Author Description** <Enter Document Name> <Enter

Author>

<Enter Document Description>

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

***Preface***

The purpose of this document is to < Detail the purpose of this document>

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

**Table of Contents**

**INTRODUCTION 6**

**TECHNICAL ENVIRONMENT 7** 2.3.1 Two types of communications in ROS 8

**Overall Architecture 9**

**SYSTEM ARCHITECTURE 9 4.3 FUNCTIONALITIES/FEATURES 13 *4.3.1 Motion Capture 13*** 4.3.2 Mimic 13 4.3.3 Relay Capture 14 4.3.4 Simulation 14

**DATABASE ARCHITECTURE 14**

**IMPLEMENTATION INSTRUCTIONS 14**

**NON-FUNCTIONAL SPECIFICATIONS 14**

**SUMMARY OF TEST RESULTS 14**

**KNOWN ISSUES & RISKS 14**

**OTHER CONSIDERATIONS 15**

**APPENDIX 15**

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

**1 Introduction**

<Paragraph summary of the technical solution that was completed including:

● Brief project description

● Brief description of technical environment

● Estimated level of complexity

● Estimated benefits/Problems it solved (quantitative & qualitative)*.>*

**1.1 Brief Project Description**

The idea of the project is to research and develop a functionality for Robot Rosie of Virtual Experience Laboratory so that Rosie can mimic human body movement. Rosie will be able to retrieve data from a human skeleton sensor; in this case it will be the Microsoft Kinect SDK. Then Rosie will be able to recreate the movement of human by using the data retrieved from the sensor.This project will allow the Baxter robots from both campuses to behave as human avatars. They will help convey the body language of the speakers from one side of the telepresence call to the other.

**1.1 Brief Description of Technical Environment**

The technical environment involved in this project are as follows:

- *Baxter Unit:* Baxter robot was manufactured by Rethink Robotics for industrial

purposes.

- *Microsoft Kinect:* It is an advanced sensor designed to capture 3D images with high

performance along with voice and face recognition.

- *ROS :* Robot Operating System (ROS) is a meta operating system which is open source that can provide functionalities implementation, package management, hardware abstraction, device control and message passing. It also gives the advantage of providing tools and libraries to write, build and run code.

**1.3 Estimated Level of Complexity**

The estimated level of complexity of the project is very high, due to the amount of knowledge required in regards to the size of the Operating System (ROS) as well as the mathematical principles related to Inverse Kinematics (getting the robot arms to move from camera input). However, there is base code that can be built on, supplied by the previous group that attempted the project.

**1.4 Estimated benefits/Problems it solved**

● Test the idea of avatar

● Allow Rosie to imitate human body movement (Microsoft kinect based arm movement)

● Implementation of docker containers **--portability**

● to run the Gazebo simulator **--testing**

● Enable communication between sensors and Rosiesti

● Send and retrieve data

● Handing retrieved data and turn it into actual robot movement

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

**2 Technical Environment** <Detail the technical environment/technologies used to complete this project and reason for the choice of the specific technology that is used. For instance, if PHP is used, why it is chosen and if GITHUB etc is used for source control please specify those details as well..>

**2.1 Baxter Unit**

Baxter unit contains distinctive sets of sensors on it to sense objects as well as people nearby which gives it the ability to attune to its surroundings. Baxter provides edge over traditional robots as it is not required to be restrictive during usage and students can work in a classroom environment. Baxter is equipped with sensors in its hands and around its arms to detect and adapt to the environment which enables it to sense any possible collision events beforehand.

Figure 2.1.1: Baxter robot

**2.2 Microsoft Kinect**

Microsoft Kinect is highly sought in robotics due to its unconventional human-robot interaction. Its components include depth sensor, microphone, set of advanced software to capture motion and gestures, motorized pivot, RGB color camera. Microsoft offers a SDK to program the Kinect sensor and integrate the sensor on the robot. Using the Microsoft Kinect SDK, various operations such as skeletal tracking, audio / image recognition, depth data etc can be performed on the robot.

Figure 2.2.1: Microsoft Kinect

**2.3 Robot Operating System (ROS)**

ROS is developed with peer-to-peer network topology being the basis. The processes in ROS are called as nodes which can run on different machines performing the computations of the system. In contrast to other robot systems which provide set only of libraries for communication and protocol, ROS provides an inbuilt system where the nodes are developed with an intention of being able to run a system on ROS with many independent modules.

The communication between these nodes happens in the form of message passing. Within the ROS build system, nodes are been developed. In the ROS framework, to send messages the nodes can utilise two different types of communication. This ROS build system is built on top of a CMake file which performs builds that are modular of both nodes and the messages that are passed between those nodes.

In addition, to provide the basis for robot control ROS incorporates many algorithms, systems and tools in the form of suite which comprises features such as:

● Mapping

● Cloud Interpretation

● Simultaneous localisation

● Vision processing algorithms

***2.3.1* Two types of communications in ROS**

Services is synchronous type of communication implemented in ROS. These are similar to function calls that are implemented in traditional programming languages. The definition of the services include the name of the string and a pair of messages which includes a request and a response.

Topics is asynchronous type of communication implemented in ROS. They are object streams published by a node. The other nodes also known as listeners has the ability to register a handler function that is called every time a new topic becomes available by subscribing. In contrast to the services, Listener nodes do not have the ability to use their subscription to the topic to communicate to the publisher. Publishing and/or subscribing to same topic can be done my multiple nodes and multiple topics can published and/or subscribed by single node.

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

**2.4 Python for Coding**

Python is chosen and more preferable to code and build on ROS over C++ because it enabled speed in to building a prototype and is a powerful language with numerous libraries. It helps in creating working demo of node faster by taking care of a lot of things and final code of node would be simple to read and understand its working/functionality. ROS Python APIs handles complex functionalities and can be accessed with effortless calls. It is uncomplicated and less time consuming to catch bugs. Python allows user to work on complex functionalities with just few lines of code.

The build system for Python modules in ROS is also much simpler than it’s C++ counterpart. This backs up the idea of how the project is run (agile; with features been created, tested and implemented quickly).

**2.5 Git for Source Control**

Git was the preferred *Version Control System* for the project, with Github being used as the remote server where the repository was stored. Git is a distributed *Version Control System*, meaning the entire codebase is shared across everyone’s machine, including the branches and history. This means that anyone anywhere can work on any part of the codebase without having to necessarily be in the VxLAB. This being the case works well for our group, as we are mostly off site except for once or twice a week.

**3 System Architecture**

**4.1 AVATAR Architecture**

The ideal architecture is two-way architecture which is symmetric in structure and can establish a remote connection between two different baxters and two kinects through a networking solution.

The AVATAR architecture can be tested in two ways using one-way architecture which is asymmetric in structure. The first test use case is where kinect does not exist and instead of kinect we use test data along with baxter to perform operations.

In the second use case is where baxter does not exist and instead of baxter we use simulator along with the kinect to perform operations.

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

**4.2 ROS Architecture for tele-operation**

ROS allows the robot control through messages which is done with the help of abstraction of capabilities from individual robots. Furthermore, individual ROS nodes which provide these capabilities can also be controlled(start/stop). ROSBridge presents integrated view to the user of the robot and its environment by encapsulating these two features of ROS.

ROSBridge primarily utilizes web sockets, both inside the robot (server) and the application (client) such as a web server for the users to interact with using JSON API. It also furnishes the user control over environment parameters and execution of ROs nodes. ROSBridge has two parts:

(1) ROSBridge Protocol to specify and send commands to ROS based on JSON. (2) ROSBridge Implementation using rosbridge\_suite which contains a collection of packages such as rosbridge\_library, rosapi, rosbridge\_server which provides transport layer for web socket and to implement the ROSBridge Protocol.

The drawback is that the robot server might a public IP address where the client accesses it via Internet or Port Forwarding, which is not really efficient as latter one is network dependent.

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

Figure 4.2.1: ROS bridge architecture

ROSLink protocol is initially developed to integrate the ROS enabled-robots with the IoT (Internet of Things). ROSLink protocol is inspired by MAVlink protocol which is a light-weight messaging protocol used for communication between drones. MAVlink protocol follows publish-subscribe to topics is data is sent as streams along with point-to-point transmission. ROSLink considers two clients, one installed in the robot and the other in the user machine. ROSLink messaging protocol is used for communication through a public server.

Fig 4.2.1: ROSLink System Architecture

**4.2.1 Components of ROSLink**

1. *ROSLink Bridge:* It acts as an interface between ROS and ROSLink protocol. The

primary functionalities are:

(1) Reading ROS topics and services messages, data serialisation in JSON

format, forwarding it to client application (2) Receiving serialised JSON data from client application, deserialize it

command parsing and execution through ROS.

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

2. *ROSLink Proxy and Cloud:* It plays the role of a proxy server between the user client application and ROSLink Bridge which is embedded in the robot. It connects by forwarding messages from ROS enabled robot to a user client application using ROSLink Bridge. It also forwards control commands given by user to the robot and updates the user with current status of the robot.

3. *ROSLink Client Application:* It takes care of robot application control and monitoring. Through the ROSLink messages it receives from the robot via the ROSLink Proxy, it monitors the status of the robot. Further, to control the activities of the robot it can also send command by utilising the ROSLink messages.

<Detail the system that was built/completed. Explain each component thoroughly. A architecture diagram is essential. >

**4.3 FUNCTIONALITIES/FEATURES** <Detail the individual/specific functionalities that comprise the system.>

motion capture, mimic, relay, simulation

***4.3.1 Motion Capture***

● Microsoft Kinect 2.0 generates image data in the form of point cloud that is published on /points topic in ROS

● The body\_tracker module of the cob\_people\_perception package converts the point cloud data into body tracking data (coordinates of user joints in 3d space relative to the sensor) that is published on the /tf topic.

***4.3.2 Mimic***

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

1. User Tracking

1. tf\_user\_tracker tracks a single user by user id from the cob body tracking

data captured from /tf 2. tf\_user\_tracker publishes the user id on the /user\_tracking topic, tf\_listen\_socket\_pub filters the cob data from /tf using this id 3. tf\_listen\_socket\_pub captures cob body tracking data captured from /tf

converts it into joint angle data corresponding with each of a baxter units arm joints. 2. tf\_listen\_socket\_pub publishes this data on a \*socket topic corresponding to each

arm joint. The rosbridge server subscribes to each of these topics and receives data from each of them 3. rosbridge server republishes the data to another client ros instance on topics

corresponding to each baxter unit arm joint. tf\_listen\_v8\_controller subscribes to each of these topics 4. tf\_listen\_v8\_controller captures data from these joint data topics and applies these

directly to rosie’s arm joints.

***4.3.3 Simulation***

● Docker simulation container can be used to simulate a real baxter unit and/or the motion capture portions of the project.

● Baxter unit is simulated in the gazebo simulator software for ROS

● A ROS utility called rosbag can be used to record and play back the /tf topics containing cob body tracking data to be reused later in a testing scenario.

● A ROS package called time\_jump can timestamp old recorded /tf data and republish it so it is not ignored by the ROS master node.

**4 Database Architecture**

There wasn’t any requirement of a database as there was not much of data that is managed

**5 Implementation Instructions**

Refer to Development Guide

**6 Non-functional specifications**

● Low Latency - the robot should be able to mimic the body movement without much time delay

● System shall work in a variety of configurations, baxter-to-baxter, baxter-to-simulator, simulator-to-simulator etc.

**7 Summary of test results**

Refer to Test Management Document

**8 Known Issues & Risks**

Refer to Issues & Risk Register

PROJECT PROPOSAL – TECHNICAL SOLUTION DESIGN, ROBOT BODY LANGUAGE TELEPRESENCE

**9 Other Considerations**

Extending the scope of this project by implementing the ideal architecture, thereby deploying the project at vietnam campus to connect the baxter to rosie remotely.