|  |
| --- |
| United States Military Academy |
| User Manual |
| Joint Cooperative Unmanned Systems Initiative |
|  |
| **Version 1.1** |
| **1/5/2012** |

|  |
| --- |
| This document provides an overview of the JCUSI project and instructions for using and troubleshooting the system. |

Table of Contents

[1. Introduction 2](#_Toc323754804)

[*1.1* Purpose 2](#_Toc323754805)

[*1.2* Problem 2](#_Toc323754806)

[*1.3.* Scope 2](#_Toc323754807)

[*1.4* Reference Material 2](#_Toc323754808)

[*1.5* Definitions and Acronyms 2](#_Toc323754809)

[2. System Overview 3](#_Toc323754810)

[*Scenario:* 3](#_Toc323754811)

[*Design:* 3](#_Toc323754812)

[3. Setting up the Network 4](#_Toc323754813)

[PICO + Robot 4](#_Toc323754814)

[Robot + OCU 4](#_Toc323754815)

[OCU+JCC 4](#_Toc323754816)

[4. Starting the System 4](#_Toc323754817)

[Initial Radio configuration 4](#_Toc323754818)

[Establishing the PICO and robot session 5](#_Toc323754819)

[5. Running the System 5](#_Toc323754820)

[Interaction with the Robot 5](#_Toc323754821)

[Explanation 5](#_Toc323754822)

[Starting the GUI 5](#_Toc323754823)

[Using the GUI 6](#_Toc323754824)

[Starting Streaming Video 7](#_Toc323754825)

[JCC Server Startup 7](#_Toc323754826)

[Command Center GUI Startup 7](#_Toc323754827)

[6. Other Actions 7](#_Toc323754828)

[DB initialization 7](#_Toc323754829)

[Appendix 8](#_Toc323754830)

# Introduction

## *1.1* Purpose

###### The purpose of this document is to describe and explain the JCUSI system for the user. A moderately technical person should be able to use this document to start and run the system. This document is a reference to the various functionality of the system.

## *1.2* Problem

###### Conducting military missions risks lives and requires manpower. The military has begun employing remotely controlled assets, especially in high risk scenarios, in order to reduce the cost of conducting operations. However, current systems still require personnel to control each system. The problem now posed is reducing the manpower for operations. How can we employ autonomous or semiautonomous systems to shift the ratio from 1 or more operators for each asset to many assets per operator?

## *1.3.* Scope

###### JCUSI narrows this problem to conducting military tasks through an autonomous or semiautonomous network of land, sea, and air assets, focusing initially on developing a system that can conduct point reconnaissance.

###### The West Point system for 2011-2012, (the current system) includes a joint command center, a service command center, and a land platform.

###### Version 1.0 includes:

###### End to end communication from the PICO to the JCC using the Packbot antenna

###### State updates from the Robot to the database and GUI on the JCC

###### Communication from the SCC to the PICO

###### Video feed from the robot to the PICO\*

###### Manual control of the robot through the PICO\*

### 

## *1.4* Reference Material

###### SRS, Version 2.0. Updated 7 DEC 2011 by CDTs Bucher, Fernandes, Liu, Lough, Nelson.

###### SAD, Version 1.1. Updated 7 OCT 2011 by CDTs Bucher, Fernandes, Liu, Lough, Nelson.

###### SRS, JCUSI Project 2010-2011. Published 14 October 2010 by CDTs Wheeler, Campbell, Martens, Ryan, Wei, Rice, Burkhardt, Taylor, Garcia-Menocal

## *1.5* Definitions and Acronyms

###### JCUSI: Joint Cooperative Unmanned Systems Initiative

###### USMA: United States Military Academy

###### USNA: United States Naval Academy

###### USAFA: United States Air Force Academy

###### SRS: System Requirements Specification

###### SAD: Software Architecture Documentation

###### JCC: Joint Command Center

###### SCC: Service Command Center

###### PackBot: A robot platform developed by iRobot

###### Aware2: The framework and architecture used to run the PackBot.

###### PICO: A scaled down PC

# System Overview

### *Scenario:*

###### JCUSI tailors its efforts around a specific scenario. In this scenario a target approaches by water, lands, and begins moving overland. The assets identify and track this target and the land and sea assets attempt to intercept and surround the target. In order to accomplish this scenario, the land portion task is primarily point reconnaissance. In order to conduct this reconnaissance, the robots receive a series of waypoint from a command center and moves along these waypoints using obstacle avoidance algorithms. Additionally, the robot provides a camera feed and has a level of autonomous image recognition in order to identify the target. The robots also retain a manual override function, because our goal is to reduce the manning needs, which does not preclude the possibility of manual control for a more flexible system.

### *Design:*

The JCUSI system is a hierarchical system consisting of a Joint Command Center (JCC), Service Command Centers (SCC), and various platforms. The JCC provides a common operating picture. One or many SCCs can be connected to the JCC through a router. These provide local control of their service’s platforms. The platforms communicate with the SCCs and operate autonomously or semi-autonomously.

# Setting up the Network

## PICO + Robot

Use a cat5 cable to connect the PICO to the robot. On the robot, the cat5 cable should plug in to the Paybreak which is plugged in to one of the payload ports.

## Robot + OCU

This connection is a wireless connection and must be reestablished every time the OCU is turned on (however, a restart will not break the wireless connection)

## OCU+JCC

Use a Cat 5 cable and the USB Ethernet NIC to connect the OCU to a router and a Cat 5 cable to connect the router to the USB. See OCU Network Configuration in Section 4 to configure you interface.

If only one SCC is being used, a crossover cable can replace the router (but you still must use the USB Ethernet NIC), alternatively, a wireless connection between the JCC and OCU is acceptable.

# Starting the System

Once the physical network is established (everything is plugged in properly), use the following steps to start the system. Some PICO modules do not need the OCU to run. Some SCC modules do not need the JCC to run. In these cases, the entire startup process is not necessary.

## Initial Radio configuration

* Power on both the robot and the OCU.
* From the OCU logon screen\*:

click options > run xclient script (options is in the bottom left of the screen)

* Login using:
  + username irobot
  + password irobot
* Select: “Just this session”
* Click “Select robot”
* Click “Select radio”
  + Select the robot ID that you want to connect to, such as “Robot id:22306”
  + >Leave channel as a default
* Once the connection is established, use the controller:
  + Press 10 (start)
    - Select quit
    - logout
  + Shutdown
* Logon using the default session (Gnome):
  + Username: irobot
  + Password irobot

\* The logon screen has a limited time to select a session. If the machine starts a Gnome session, you can log out and continue to follow the instructions as before.

## Establishing the PICO and robot session

* Turn on the PICO.
* From the PICO
  + To enable packet forwarding, open a terminal use the following commands:
    - ssh to the robot, such as “ssh root@172.16.87.34” (the password is packbot)
    - cd /
    - python ipforwardv3.py
    - exit
    - You can close the terminal
* Now the iptables for packet forwarding are configured
* To end the session that was established to the radio
  + Open Firefox
  + Navigate to:
    - http://172.16.87.34/robot/releaseSession.html?session=ThirdPartyFasTacSession

# Running the System

## Interaction with the Robot

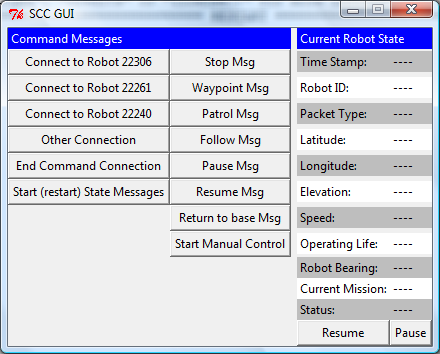
### Explanation

The SCC GUI controls the robot. The robot must be ready to accept a session before you connect it.

### Starting the GUI

* Open a new terminal
* Change directory to /Desktop/Version1.1/Code/SCCcode/
* Run “python SCCGui\_withState.py”

### Using the GUI



* Connect to X
  + These buttons require the robot to be already up and listening. They create a connection on port 52000.
* Other Connection
  + Same as Connect to X but allows the user to specify the host and port
* End Command Connection
  + Tells the robot to kill the connection and closes the socket on the SCC side.
* Start (restart) State Messages
  + Sends a message to the robot to start sending state messages and which port to send them to. Each click increments the port number to overcome the bind issues.
* Stop Msg
  + Sends a message to the robot that orders it to stop the execution of current commands.
* Waypoint Message
  + Send the robot a point to navigate to with decimal notation latitude and longitude.
* Start Manual Control
  + Allows the user to send direct commands to control the robot chassis from a terminal.
  + ‘exit’ ends the manual control
  + See Appendix for Robot Commands
* All other buttons send a message corresponding to their specified task but the robot is not currently configured to implement them.
* Current Robot State
  + All fields do not update 100% of the time but they continuous pull a feed from the state messages, updating the fields, unless halted by user input.
  + Pause button
    - Freezes the current state information labels as they are until the user presses the resume button.

### Starting Streaming Video

* CD to /RTP/VideoExamples/
* “./ VideoExample -p 172.17.XXX.XXX”

## JCC Server Startup

* On the JCUSI05 or 06 laptop, open the directory for the JCCcode
  + In a command window type: python JCCserver.py
  + The terminal should run the server and display a listening message
* WARNING: If you run messages through the JCC and attempt to update the database when the robotID is not already registered it will fail.  To ensure that the robot is registered, run the “Register Robot” command from the GUI and plug the robot id in.

## Command Center GUI Startup

* On the JCUSI05 laptop, open the Version1.1/Code/Command Center 3 directory
* Click on Command\_Center\_3.sln to open in Visual Studio
* Click the run command and the GUI will display.
* If the Google Map does not display properly, check the wireless connection and restart the GUI

# Other Actions

## DB initialization

* To create the database on your machine and initialize it with some established target and robot state data, open MySql.
* Open the commandcenter\_db.targets….sql script and execute all.
* This should create the database on the localhost with several robots.

## Appendix

|  |  |  |  |
| --- | --- | --- | --- |
| FasTac Robot Commands | | | |
| Command | **Action** | **Command** | **Action** |
| 1 | Open the Gripper | w | Cam Arm Up |
| 2 | Close the Gripper | s | Cam Arm Down |
| 3 or 4 | Rotate Gripper | t | Zoom in |
| 5 or 6 | Pan the Camera | g | Zoom Out |
| 7 or 8 | Tilt the Camera | y | Aggressive Stance |
| 9 | Storage Position | h | Travel Stance |
| 0 | Stop All Arm Actions | i | Drive Forward |
| l | Lights | j | Left Turn |
| , or . | Flipper Movement | k | Right Turn |
| z | Stop Movement | m | Drive Backwards |
| b | Brake |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| EOD Robot Commands | | | |
| Command | **Action** | **Command** | **Action** |
| 1 | Open the Gripper | w | Shoulder Up |
| 2 | Close the Gripper | s | Shoulder Down |
| 3 or 4 | Rotate Gripper | q or a | Rotate Turret |
| 5 or 6 | Pan the Camera | e | Elbow 1 Up |
| 7 or 8 | Tilt the Camera | d | Elbow 1 Down |
| 9 | Storage Position | r | Elbow 2 Up |
| 0 | Stop All Arm Actions | f | Elbow 2 Down |
| l | Lights | p | Aux Camera On |
| , or . | Flipper Movement | o | Aux Camera Off |
| z | Stop Movement | l | Lights |
| + | Zoom in | c | Close Blast Shield |
| - | Zoom Out | v | Open Blast Shield |
| y | Aggressive Stance | b | Brake |
| h | Travel Stance | , or . | Flipper Movement |
| i | Drive Forward | [ | Secondary Camera 1 |
| j | Left Turn | ] | Secondary Camera 2 |
| k | Right Turn | { | Secondary Camera 3 |
| m | Drive Backwards | } | Aux Camera |