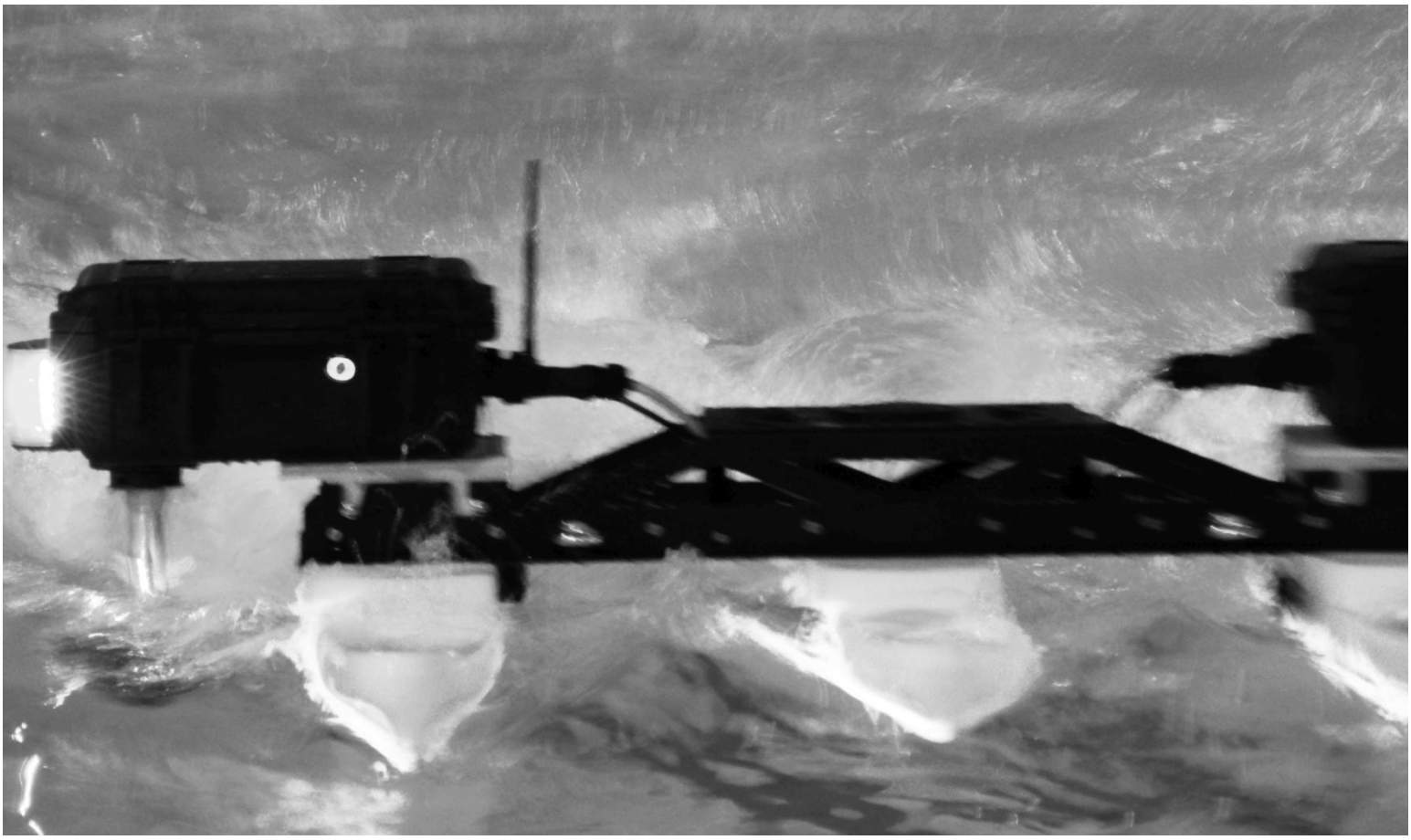
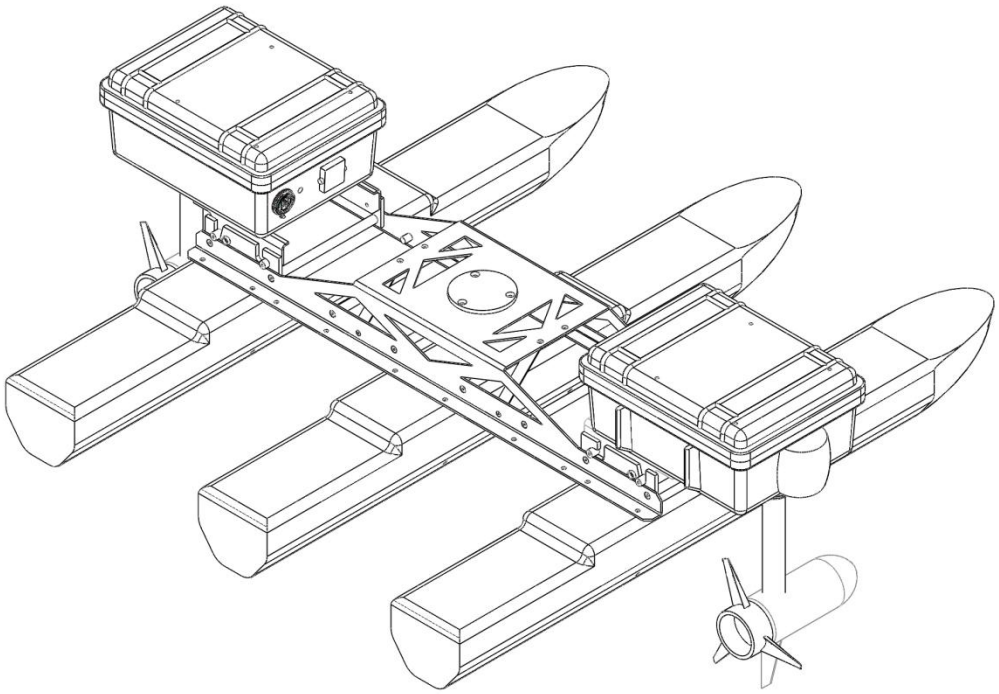
CLEARPATH ROBOTICS™

KINGFISHER M100™

UNMANNED SURFACE VEHICLE



USER MANUAL



CONTENTS

INTRODUCTION

Clearpath Robotics Kingfisher M100 is a portable, agile, and easy-to-use unmanned surface vehicle for rapid prototyping applications. In this guide, you will find information about the setup, operation, and maintenance of your Kingfisher M100.

What's Included

Included with the base Kingfisher M100 kit are the following:

- 1× Clearpath Robotics Kingfisher M100 Main Frame
- 1× Primary Thruster Module, including FitPC2 and GPS module
- 1× Secondary Thruster Module, including VIP Series wireless radio
- 2× 12V NiMH Battery Packs (pre-installed in Thruster Modules)
- 2× NiMH Battery Chargers

What's Required

The onboard FitPC2 included in Kingfisher communicates with the Kingfisher's microcontroller using the Clearpath Control Protocol (CCP) over RS232 serial. This computer has been set up with Ubuntu Server, Robot Operating System, and the Clearpath Control Protocol interfaces for C++ and Python.

To use Kingfisher M100, you will require an external PC running Windows or Linux. Depending on your package configuration, a suitable machine may have been included and pre-configured.

THE BASICS

This section provides an overview of the key specifications of the Kingfisher M100 platform. Figure 1 gives a tour of important Kingfisher M100 components.

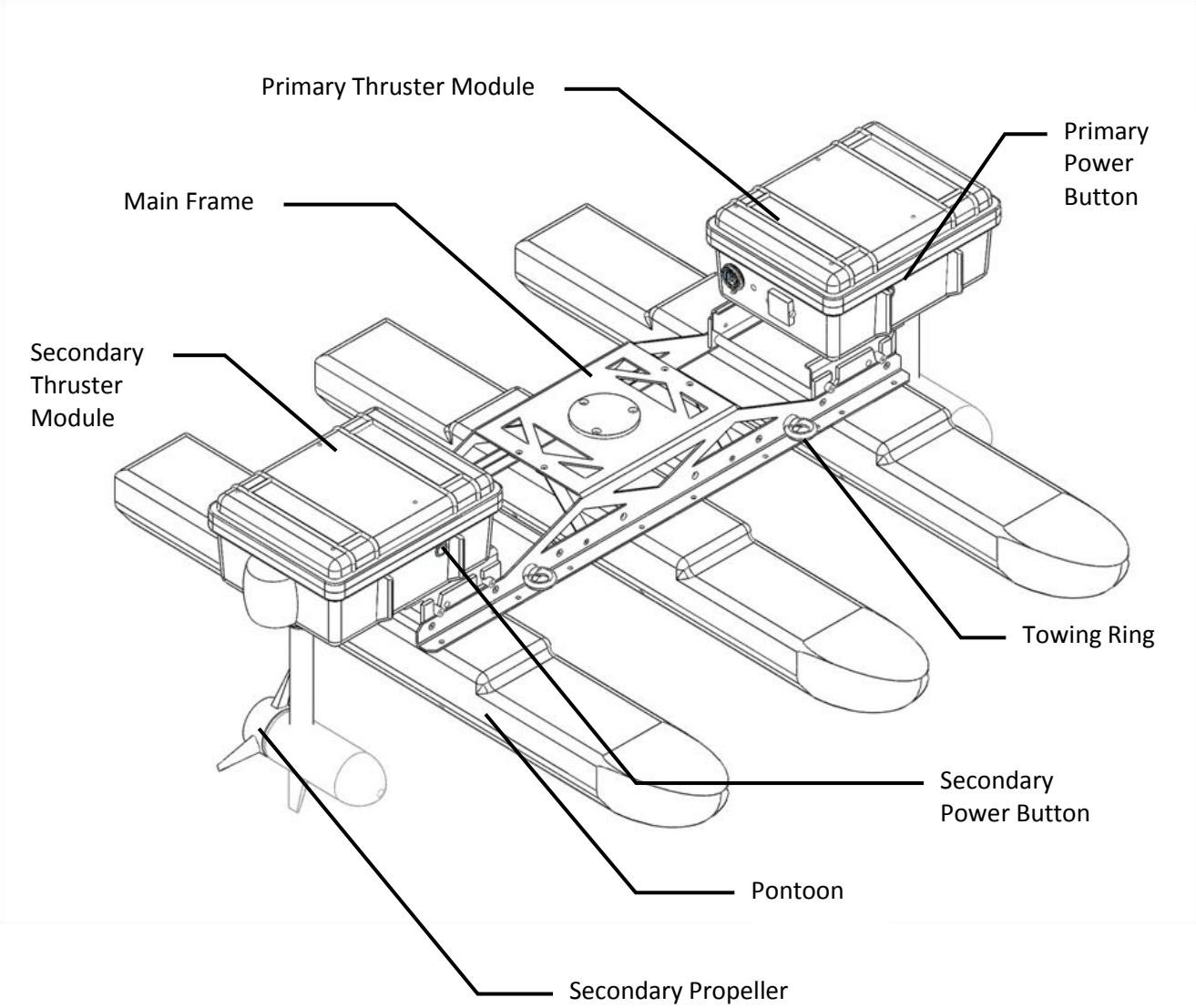


Figure 1: Kingfisher M100 at a Glance

Figure 2 provides an overview of the insides of the primary and secondary thruster modules.

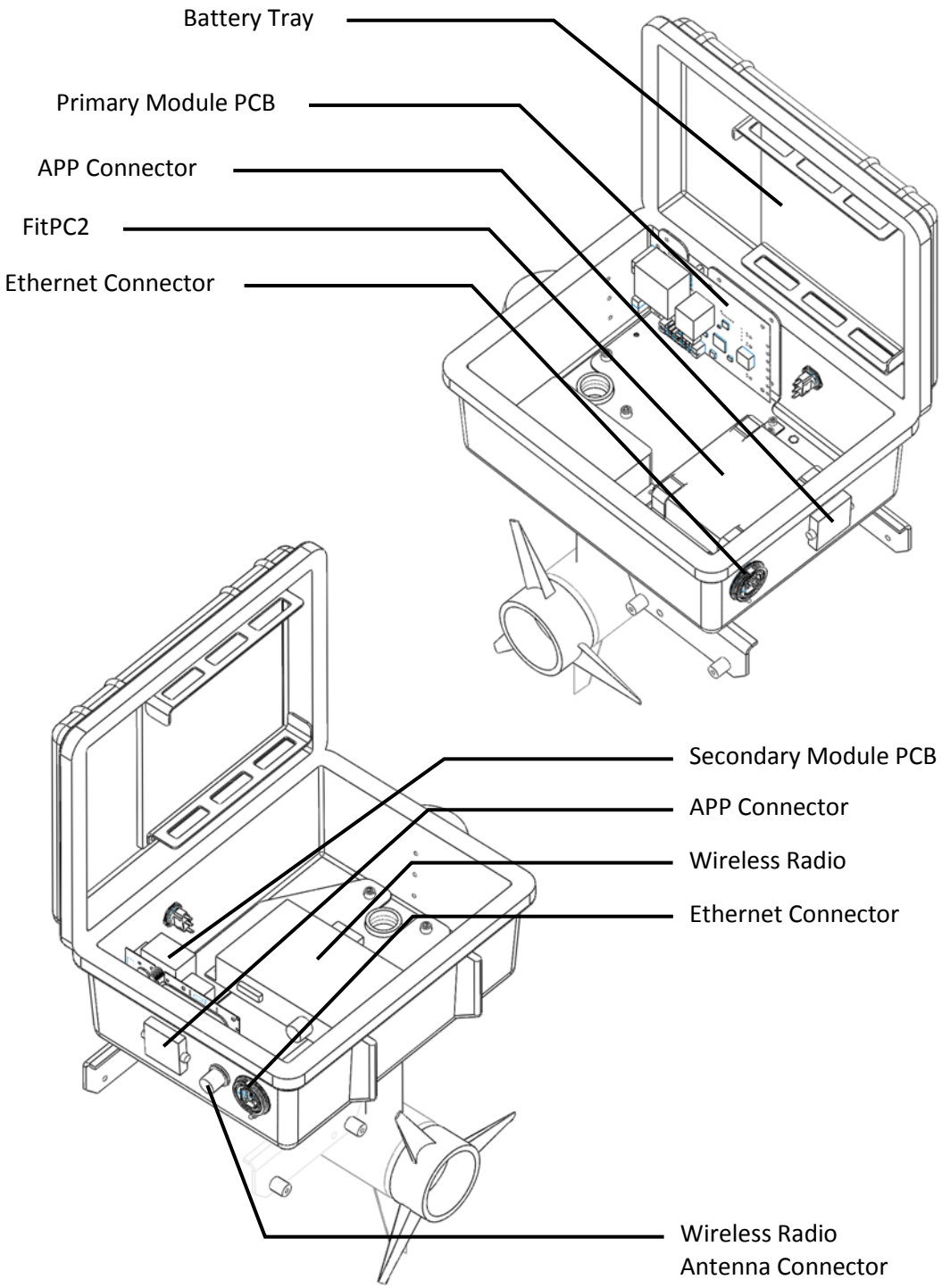


Figure 2: Thruster Modules

Figure 3 gives a overview of the Clearpath Robotics Mobile Base Station.

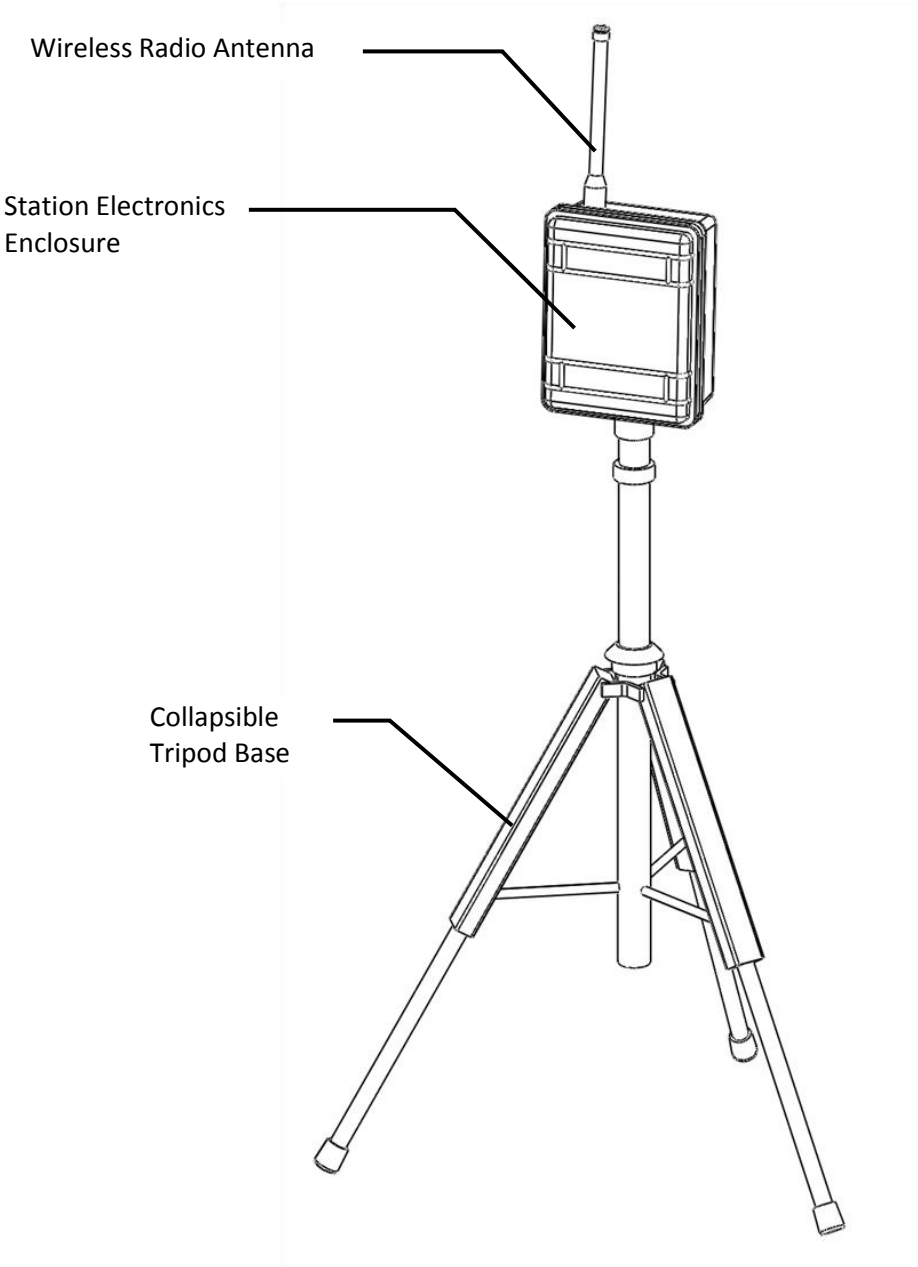


Figure 3: Mobile Base Station

Orientation References

The reference frame used by all Clearpath Robotics vehicles is based on ISO 8855, and is shown in Figure 4. Kingfisher is being viewed from the front; when commanded with a positive translational velocity (forward), wheels travel in the positive x-direction.

The direction of the axes differs from those used for roll, pitch, and yaw in aircraft, and care should be taken to ensure that data is interpreted correctly.

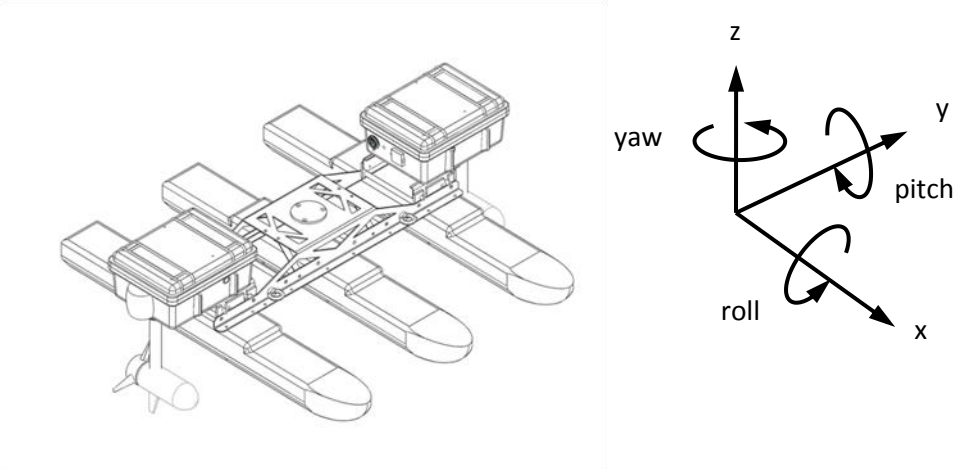
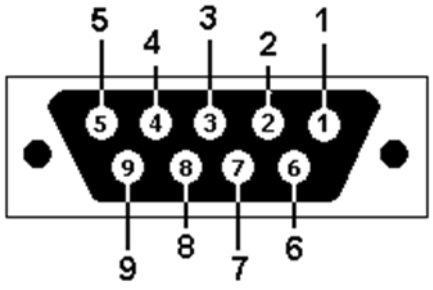


Figure 4: Kingfisher M100 Reference Frame

Pinout References

Kingfisher M100 provides a female DE-9 connector, for communication with the onboard FitPC2. The pinout of this connector is shown in Figure 5.



Pin	Name	Dir	Description
2	RX	IN	Data from Platform
3	TX	OUT	Data to Platform
5	GND	N/A	Common Ground

Figure 5: Kingfisher M100 DE9 Pinout

System Specifications

Key specifications of Kingfisher M100 are shown in Table 1.

Dimensions	1270 mm length 1270 mm width 520 mm height	50 in length 50 in width 20.5 in height
Weight	30 kg	66 lbs
Payload (rated)	5.7 kg	11.5 lbs
Speed (max)	1.3 m/s forward 0.7 m/s reverse	3.0 mph forward 1.7 mph reverse
Draught	300 mm	12 in
Thrust	270 N	60 lb-f
Turn Radius	None	
Operating time	4 hours typical 8 hours maximum	
Drive Power	200 W peak	
Battery	2× 12V 14 Ah NiMH	
Battery charger	Short-circuit, over-current, over-voltage, and reverse voltage protection.	
Charge time	9 hours	
System Interface	RS-232 Serial, 115200 Baud	
Radio Interface	Encrypted 2.4 GHz, 54 mbps Broadband 2.4 GHz DSS Remote Control	
MCU sensing	Battery status Motor current Compass	
Additional Sense	GPS	

Table 1: Kingfisher M100 System Specifications

Control Modes

At the microcontroller level, Kingfisher M100 offers direct control over the PWM output to the primary and secondary propellers.

- **Voltage control**, accessed via CCP message **0x0202**, allows the direct control of motor voltage, specified as a percentage of 12V.

Speed or position control may be implemented at the PC controller level, using GPS, compass, and accelerometer data to provide the necessary feedback. For more details, please refer to the Clearpath Control Protocol documentation, and the documentation specific to your chosen software API.

Navigate to http://clearpathrobotics.com/downloads to download the current release of the Clearpath Control Protocol.

Monitoring

In your software, use CCP message **0x4004** to request general system status. There are five voltage and six current fields returned. The five voltages included are:

1. **Main Bus.** Main bus voltage. This is the shared voltage line used by the electronics and motors in both modules. It should approximately match whichever of the two batteries is higher voltage. It is possible—but not recommended—to operate Kingfisher with a single battery only.
2. **Primary Battery.** Voltage level of the port-side battery.
3. **Secondary Battery.** Voltage level of the starboard-side battery.
4. **Primary Motor.** Voltage level of the port-side propeller motor.
5. **Secondary Motor.** Voltage level of the starboard-side propeller motor.

The six current levels measured are:

1. **Primary Battery Current.** Total current drawn from port-side battery.
2. **Secondary Battery Current.** Total current drawn from starboard-side battery.
3. **Primary Motor Current.** Current consumed by port-side motor.
4. **Secondary Motor Current.** Current consumed by starboard-side motor.
5. **Primary Control Current.** Current consumed by port-side electronic devices, including MCU, GPS, and FitPC2.
6. **Secondary Control Current.** Current consumed by starboard-side electronic devices, including compass, accelerometer, and VIP Series wireless radio.

SAFETY

Clearpath Robotics is committed to high standards of safety. Kingfisher M100 contains several features which protect both the safety of users and the integrity of the craft.

General Warnings

Kingfisher M100 is an agile and high-performance vehicle, but the propellers are sharp and can do a lot of damage! For the safety of yourself and others, always conduct initial experiments and software development with the vehicle raised off the ground.

When starting out, favor slower propeller speeds, beginning with 10% of full power. When Kingfisher is operating, keep clear of the propellers. Have the remote control available at all times for manual takeover in case of a software malfunction.

One or both of the propellers may spin slowly when the vehicle is idling. This is normal. If extensive bench tests are being conducted and the propellers are not required, unplug the motor connectors within each thruster module.

RC Manual Takeover

The primary safety mechanism included with Kingfisher is the ability to take over operation of it at any time using the included RC unit, shown in Figure 6.

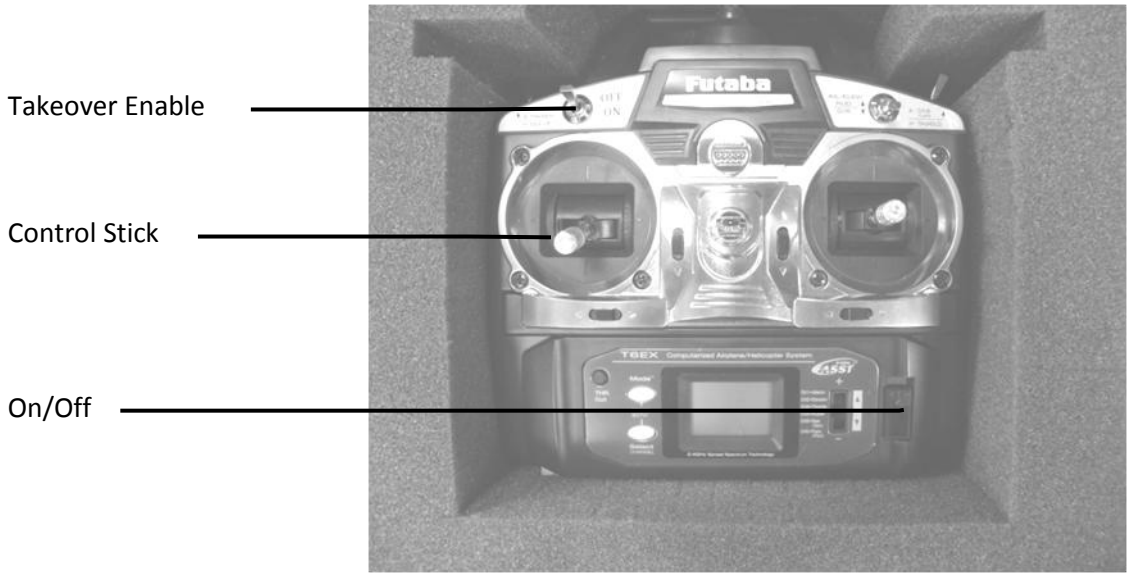


Figure 6: Kingfisher M100 Remote Control

For takeover to be successful, the RC controller must be switched on, have adequate battery power, and have the Takeover Enable switch in the `on` position. Before each operating session with Kingfisher, briefly confirm the RC takeover function.

When in manual mode, the FitPC2 will still be able to request system status information, but any motion commands issued over serial will be ignored.

Electrical System

Kingfisher is powered by dual 12V NiMH battery packs, of a similar type to those used in electric and hybrid vehicles. Please observe the following precautions regarding Kingfisher's electronic components:

- Do not tamper with the plug attached to the battery.
- Do not tamper with the thruster module circuit boards.
- Do not operate Kingfisher in the water without both modules firmly seated, latched closed, and powered on.
- Charge the battery packs only with chargers provided by Clearpath Robotics.
- Return the battery packs to Clearpath Robotics for proper disposal.

Lifting and Transport

For the safety of users and to maximize the lifetime of Kingfisher, please observe the following when transporting the robot:

- Kingfisher should be lifted by two persons, firmly gripping either the front and back of the pontoons, or the thruster module cases. **Do not lift by the propeller posts.**
- Always transport Kingfisher with the thruster modules detached from the main frame, powered off, and with the batteries disconnected internally.

GETTING STARTED

You are ready to go! This section details how to get Kingfisher and the mobile base station set up.

Base Station Setup

To set up the Mobile Base Station:

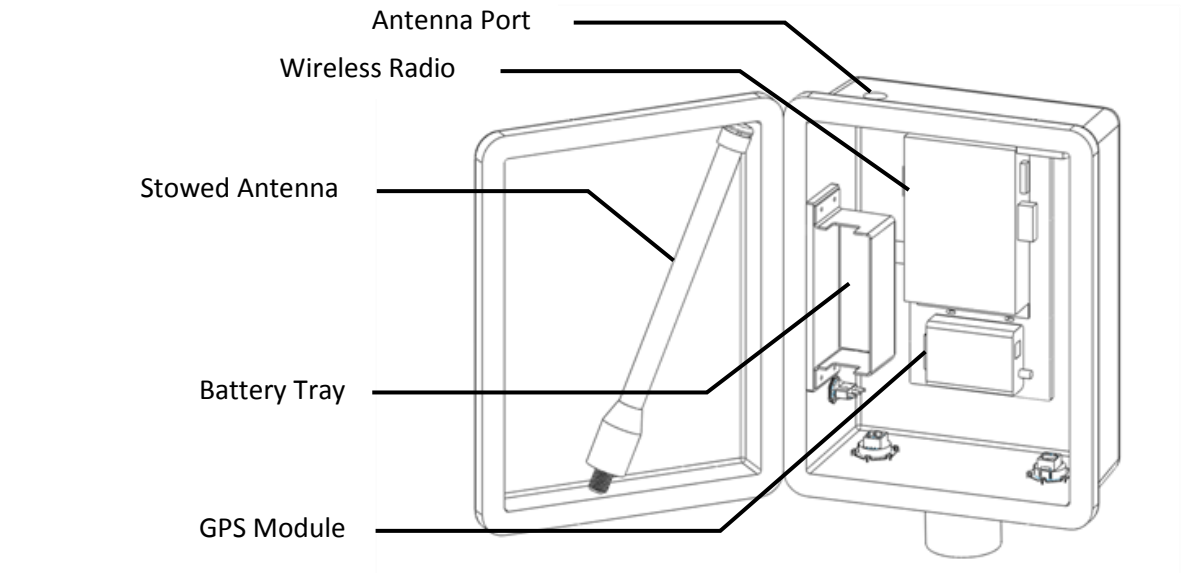


Figure 7: Mobile Base Station Enclosure

1. Open the base station enclosure case, as shown in Figure 7.
2. Remove the stowed wireless antenna from inside the lid, and screw it into mating port on top of base station.
3. Ensure that battery is present, firmly strapped into its tray, and connected to the power switch.
4. Unfold the base station tripod, and place the base station enclosure on the top of the tripod, tightening the tripod clamping sleeve as necessary.
5. Connect laptop computer to USB and Ethernet ports on bottom of base station.
6. Close and power on base station.

To test base station radio, open a terminal on the connected laptop, and type:

```
ping <base station radio ip>
```

Base station radio IP can be found on a label on the base station radio.

Kingfisher Setup

To set up Kingfisher:

1. Attach the three pontoons to the main frame, using four thumbscrews per pontoon, as shown in Figure 8.

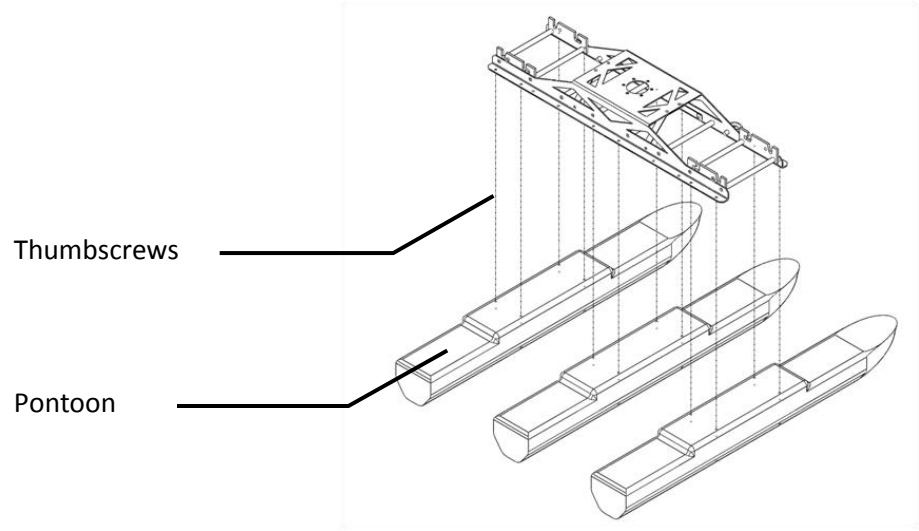


Figure 8: Pontoons to Main Frame

2. Remove thruster modules and remote control from carry case.
3. Place pontoon frame on thruster module carry case.
4. Position primary thruster module on left side of main frame, such that inner mounting pins are seated in L-shaped mounting cradle grooves, as shown in Figure 9.

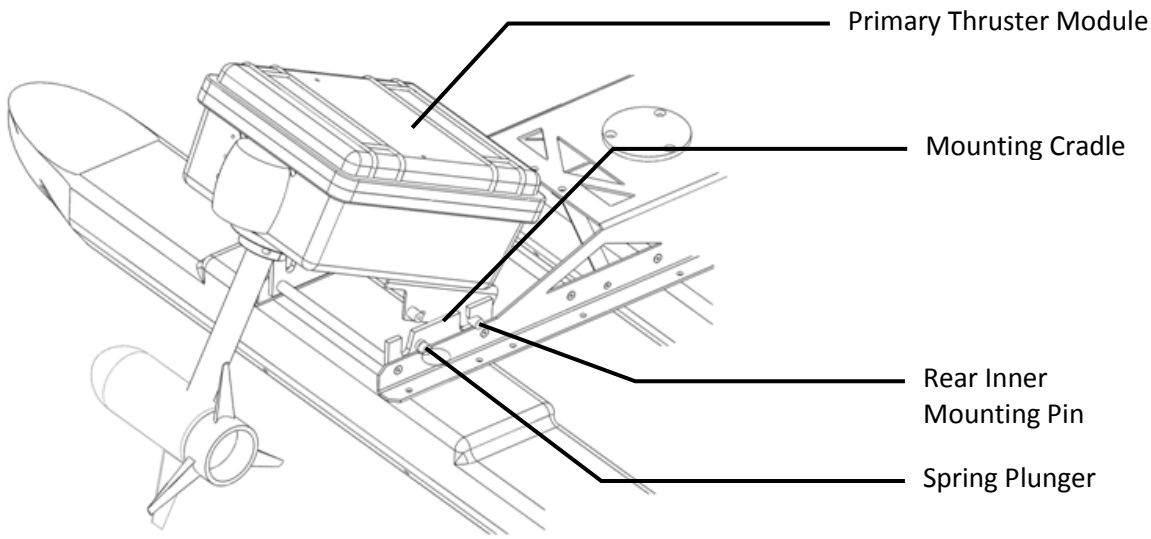


Figure 9: Mounting Thruster Modules

5. Pull spring plunger, and tilt thruster module into place.
6. Repeat procedure with secondary thruster module.
7. Connect orange APP cable to mating connectors on each thruster module.
8. Connect black Ethernet cable to mating connectors on each thruster module.

9. Screw the black “rubber ducky” wifi antenna onto the mating connector on the secondary thruster module.
10. Open thruster modules and in each, connect the battery power plug to the mating red and black power plug. Close and firmly latch thruster modules, ensuring that no cables have become pinched.
11. Power on thruster modules, and verify that the running lights flash on both. This confirms CAN communication between modules.
12. From the laptop connected to the base station, attempt to ping Kingfisher’s onboard radio:

```
ping <kingfisher radio ip>
```

13. Now, confirm communication with Kingfisher’s onboard FitPC2:

```
ping <kingfisher host>
```

14. You are now ready to perform end-to-end verification!

Verification

From the external PC, attempt to connect to the onboard PC using SSH:

```
$ ssh administrator@<kingfisher host>
```

Passwords, IPs, and hostnames can be found on a label placed on the onboard PC.

When connection is successfully made, using the Clearpath Python API to connect to the Kingfisher MCU:

```
$ python –m clearpath.horizon.cli
```

When the connection is made successfully, you should see a readout of information about your Kingfisher M100 platform, followed by a prompt where you may directly issue Clearpath Communication Protocol commands.

Ensure that the propellers are free to spin safely, and then at the prompt, issue a command to spin at 20% power forward:

```
clearpath:/dev/ttyUSB0$ set_differential_output 20 20
```

The propellers should briefly spin before timing out. For safety, Clearpath mobile platforms require a constant stream of commands at 10Hz in order to function.

To maintain the commanded speed, use the repeat command to send the propeller command continuously at 20 Hz:

```
clearpath:/dev/ttyUSB0$ repeat 20 set_differential_output 20 20
```


With the wheels spinning continuously, try using `request_system_status` to return some key data from Kingfisher:

```
clearpath:/dev/ttyUSB0$ request_system_status
```

You should see voltage and current readouts, as described in Monitoring, on page 10.

When finished, cease the repeated command and exit:

```
clearpath:/dev/ttyUSB0$ repeat off
clearpath:/dev/ttyUSB0$ exit
```

For a full list of all available commands, type `help` at the prompt. For help on a specific command, type `help <command>`, for example `help set_differential_output`.

Teleoperation

Clearpath Robotics provides an officially-supported Robot Operating System (ROS) driver for Chameleon M100, based on the Python interface. ROS is a flexible, distributed architecture for controlling robotic systems, and is a recommend method for controlling Chameleon. Using ROS, you can quickly and easily take joystick control of Chameleon.

A simple ROS teleoperation network is shown in Figure 10. The `joy_node` is a device driver which interfaces with the joystick and produces ROS **Joy** messages. The `clearpath_teleop` node consumes these **Joy** messages, applies scale factors and produces **Twist** messages. The **Twist** messages are received by `clearpath_base`, which uses CCP to forward them as output commands to Kingfisher.

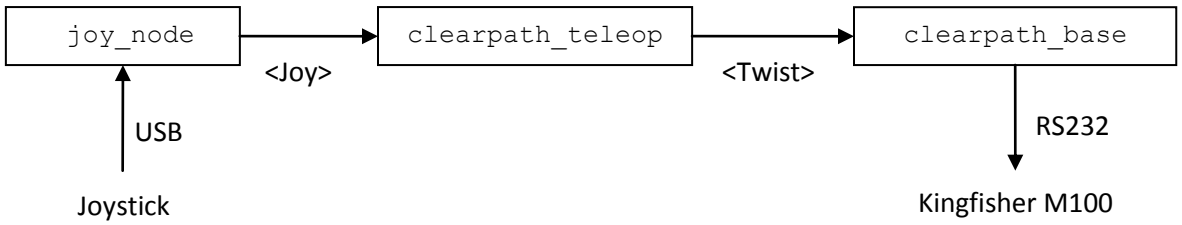


Figure 10: ROS Teleoperation Network

With a joystick connected directly to the base station, edit the teleoperation launch file:

```
vi ~/ros-clearpath/teleop_m100.launch
```

The first thing to check in this file is that the joystick section has the correct device for your plugged-in joystick:

```
    <node pkg="joy" type="joy_node" name="joy_node">
      <param name="dev" value="/dev/input/js0" />
      <param name="deadzone" value="0.3" />
    </node>
```


Confirm that your joystick is /dev/input/js0, and if not, change the launch file accordingly.

The second thing to check is the hostname on which to launch the clearpath_base node. You will want to launch this node on Kingfisher, not locally:

```
    <node pkg="clearpath_base" type="raw.py" name="clearpath_base"
output="screen" machine="<robot-hostname>">
     ...
    </node>
```

Be sure to change the robot hostname to match the hostname of Kingfisher's onboard FitPC2.

When ready, ensure that Kingfisher is safely up on blocks, and launch the teleoperation network:

```
roslaunch clearpath_teleop teleop_m100.launch
```

You should now be able to drive the robot from the connected joystick. When operation has been confirmed, carefully launch Kingfisher into the water.

For more details, consult the following pages on the official ROS wiki:

- http://www.ros.org/wiki/ROS/NetworkSetup
- http://www.ros.org/wiki/roslaunch/XML/node
- http://www.ros.org/wiki/roslaunch/XML/machine

OPERATION

This section provides a guide to using the various available control methods on Kingfisher.

Control Protocol

The Clearpath Control Protocol (CCP) is a simple way for users to interface with Clearpath hardware from a PC or other higher-level controller. All Clearpath Robotics hardware implements the subset of messages relevant to the specific chassis configuration present. For example, Kingfisher M100 provides handlers for the differential drive output message, but not for the velocity message (since Kingfisher contains no low-level velocity feedback mechanism).

CCP includes several features intended to increase communication reliability while keeping message overhead and implementation complexity low. It is not intended for multiple devices to be simultaneously connected to a single communication line, removing the need for addressing or bus negotiation.

CCP is a binary serial protocol, documented in full in the CCP Guide. Provided by Clearpath Robotics are implementations in Python, C++, and LabVIEW, as well as a node for Robot Operating System (ROS) built on the Python implementation. This section provides an overview and introduction to the two low-level implementations.

Python

The below code requests and prints a 1 Hz system status update and ramps the motor driver outputs from 0 to 100% voltage.

```
#!/usr/bin/python
from clearpath import Interface
from clearpath.transports import Serial
import time

cpr = Interface(transport=Serial,
                transport_args={'port':'/dev/ttyUSB1'})
cpr.open()

def status_handler(code, payload, timestamp):
  print(payload)

cpr.add_handler(status_handler, request='request_system_status')
cpr.request_system_status(subscription=1)

for i in range(0, 10):
  left_percent = right_percent = i * 10
  cpr.set_differential_output(left_percent, right_percent)
  time.sleep(0.5)

time.sleep(1)
cpr.close()
```


If you would prefer to poll for received messages rather than use callbacks, you may use the `get_waiting` method to receive a list of unhandled messages. For more details on using the Python driver, see the included documentation and examples.

C++

This code is similar to the Python demo.

```
#include <iostream>
#include <cstdio>  // getchar
#include <cstdlib>
#include <unistd.h>
#include <typeinfo>
#include "clearpath.h"

using namespace std;
int main(int argc, char *argv[]) {
    /* Configure the serial port */
    const char* port = (argc == 2) ? argv[1] : "/dev/ttyUSB0";
    clearpath::Transport::instance().configure(port, 3 /* max retries*/);

    /* Subscribe to some interesting data */
    clearpath::DataSystemStatus::subscribe(1);
    clearpath::DataSystemStatus * cur_status = NULL;

    /* Ramp speed to 0.6m/s over 6 seconds.
     * We use the 1Hz system status message for timing, so we want to
     * begin by waiting for the first message, which is sent shortly
     * after the beginning of the subscription */
    cur_status = clearpath::DataSystemStatus::waitNext();
    cout << *cur_status << endl;
    delete cur_status;

    for(int i=1; i<=6; ++i) {
        // Set motor speed
        clearpath::SetDifferentialOutput(0.1*i, 0.1*i).send();

        /* Wait for the next system status message to arrive */
        while(!(cur_status = clearpath::DataSystemStatus::popNext()));
        cout << *cur_status << endl;
        delete cur_status;
    }

    /* Terminate subscriptions */
    clearpath::DataSystemStatus::subscribe(0xffff);

    return 0;
}
```

Distributed with the C++ API are a few simple test programs. An applicable example is:

`botinfo.cpp`: Requests four pieces of information from Kingfisher M100 and exits

BATTERY & MAINTENANCE

Kingfisher A200 is built for long-term use. However, there are steps which can be taken to maintain and extend the life of the platform even further.

Charging

To recharge the batteries, plug them into the supplied chargers (red terminal connecting to the red terminal, black terminal connecting to the black terminal). The batteries can be charged while inside the thrusters. Avoid leaving the batteries connected to the charger for an extended period of time after charging is complete. Always charge the batteries at room temperature. The battery takes approximately 10 hours to recharge after a full discharge.

Battery Care

NiMH battery cells undergo a self-discharge rate of about 30% a month at room temperature. Storing NiMH cells at lower temperatures helps to slow this process. Storing the NiMH cells in a freezer will reduce the self-discharge significantly (provided they are kept dry). The battery cells must be warmed to room temperature before charging or use. To prevent permanent capacity loss due to self-discharge, place batteries on the charger at least once a month to top them up. When using for the first time after long-term storage, repeated charge-discharge cycles may be needed to restore the batteries to original performance.

Thruster Modules

Kingfisher's thruster modules are designed and shown in testing to be water-tight. However, if one is submerged for an extended period of time, it is recommended that it be opened up and inspected for any wet components. If any inside components have become wet, power down the unit immediately, disconnect the battery, and allow to air dry for 24 hours.

TIPS AND TROUBLESHOOTING

Tips

Kingfisher is tough, but it's also agile, and the propellers are not designed to interact with anything other than water. Exercise care when driving near walls, obstacles, docks, and shallow bottoms.

Troubleshooting

This section lists a few possible issues which may be encountered.

- **No blue light when pressing the power button.** Ensure that the battery is charged and correctly connected. Use a multi-meter to verify the voltage on the battery terminals.

- **One or both propellers is slowly spinning when the vessel is idling.** If this is a concern when bench-testing, unplug the motor connector within the applicable thruster module. If it affects autonomous operation, we suggest adding a software trim to the motor commands. If it affects RC operation, adjust the trim settings on the remote control unit.

- **Starboard running light is not blinking and starboard thruster is not responding.** Ensure that the orange APP connector is connected between the modules and either a RC module or loopback adapter is connected to the primary circuit board.

If you're having some trouble that you don't see here, or the suggested solution isn't working out, please get in touch so we can help you with it.

A. PC SETUP

This appendix details the software on a Clearpath-ready PC, and the setup necessary to prepare a computer for use with Husky.

Python

The recommended starting point for users of Clearpath Robotics products is the software interface written in Python, called clearpath-py. In order to use clearpath-py, you will need to install Python 2.6 or 3.0, and the pyserial module. Most Linux distributions come with a recent Python install, but Windows users will need to visit http://python.org/download/ to download an installer.

To verify your Python version, open a command prompt and type `python`. You should see an output similar to the following:

```
$ python
Python 2.6.5 (r265:79063, Apr 16 2010, 13:57:41)
[GCC 4.4.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

If Python is not installed, or you see a version number less than 2.6, you will need to install a newer version from the Python website, or use your Linux distribution's package manager to upgrade to version 2.6.

To check for pyserial, type `import serial` at the python prompt:

```
>>> import serial
>>>
```

If you see an error message, you will need to install pyserial, found through package management (`python-serial` in the Ubuntu repositories), or from http://pyserial.sourceforge.net/.

Navigate to http:// clearpathrobotics.com/downloads to download the current release of the Python API, and follow the directions in the README file.

ROS

To install ROS, follow the instructions listed at http://www.ros.org/wiki/ROS/Installation.

When your ROS installation is complete and verified, use Subversion to check out the Clearpath ROS packages:

```
$ svn co http://clearpath-ros-pkg.googlecode.com/svn/trunk/ clearpath-ros-pkg
```

Ensure that clearpath-ros-pkg is in your **ROS_PACKAGE_PATH** environment variable. You can test this using *rospack*:

```
$ rospack find clearpath_base
```

When *rospack* is correctly able to find the Clearpath packages, proceed to compile the message definitions:

```
$ rosmake clearpath_base
```

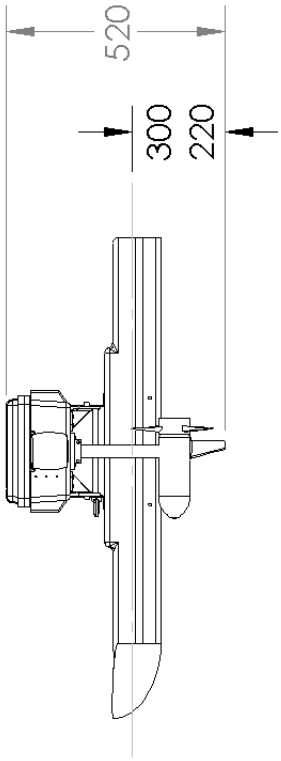
When this is complete, you should be ready to go!

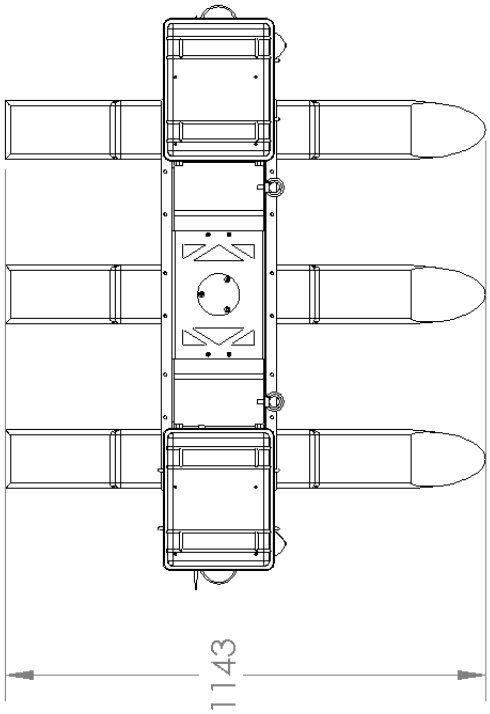
C++

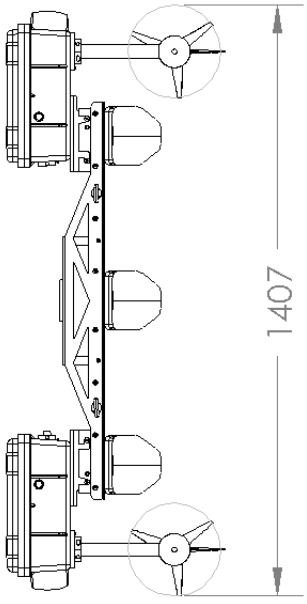
Navigate to http:// clearpathrobotics.com/downloads to access the current release of the C++ API.

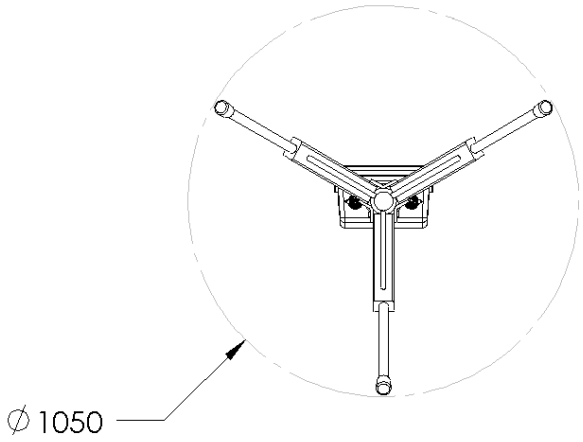
B. PRODUCT DIMENSIONS

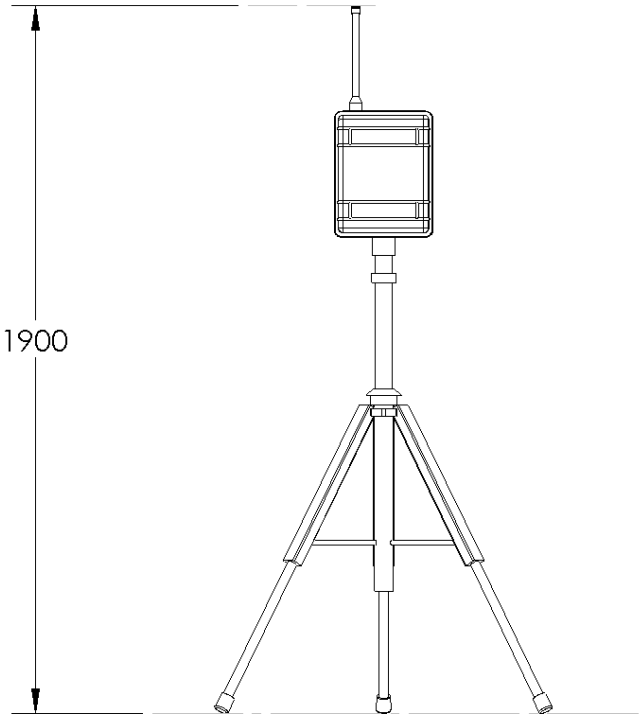
All measurements given in mm.











C. SERVICE AND SUPPORT

Clearpath Robotics is committed to your success and satisfaction. We are located in Waterloo, Ontario, and can accept phone calls from 9AM to 5PM Monday to Friday, at our toll-free number, or emails at any time.

1-800-301-3863

support@clearpathrobotics.com

