Water Resources Research
Supporting Information for

Raspberry Pi Reflector (RPR): a low-cost water-level monitoring system based on GNSSInterferometric Reflectometry

Makan A. Karegar[1*], Jürgen Kusche[1], Felipe Geremia-Nievinski[2], Kristine M. Larson

[1] Institute of Geodesy and Geoinformation, University of Bonn, Bonn, Germany
[2]Department of Geodesy and Postgraduate Program in Remote Sensing, Federal University of Rio Grande do Sul, Porto Alegre, Brazil
[3]Aerospace Engineering Sciences, University of Colorado, Boulder, USA
*corresponding author: karegar@uni-bonn.de

Contents of this file

Figures S1 to S2

Text S1: Installation guide for RPR

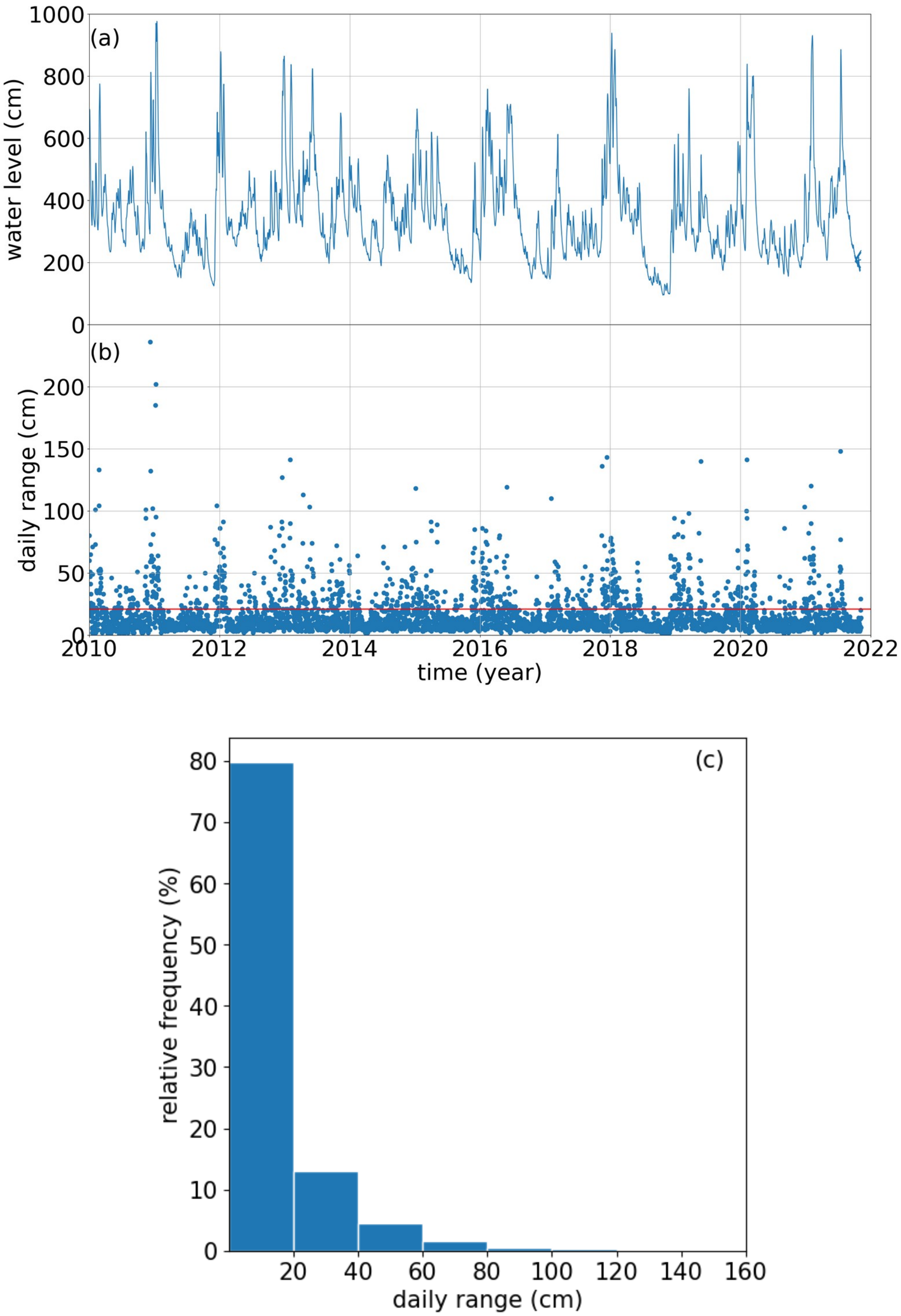

Figure S1. (a) 15-minutes records of river level from the river gauge at Wesel, Germany. **(b)** daily range values. The horizontal red line marks daily range of 20 cm. **(c)** relative frequency histogram for daily range data.

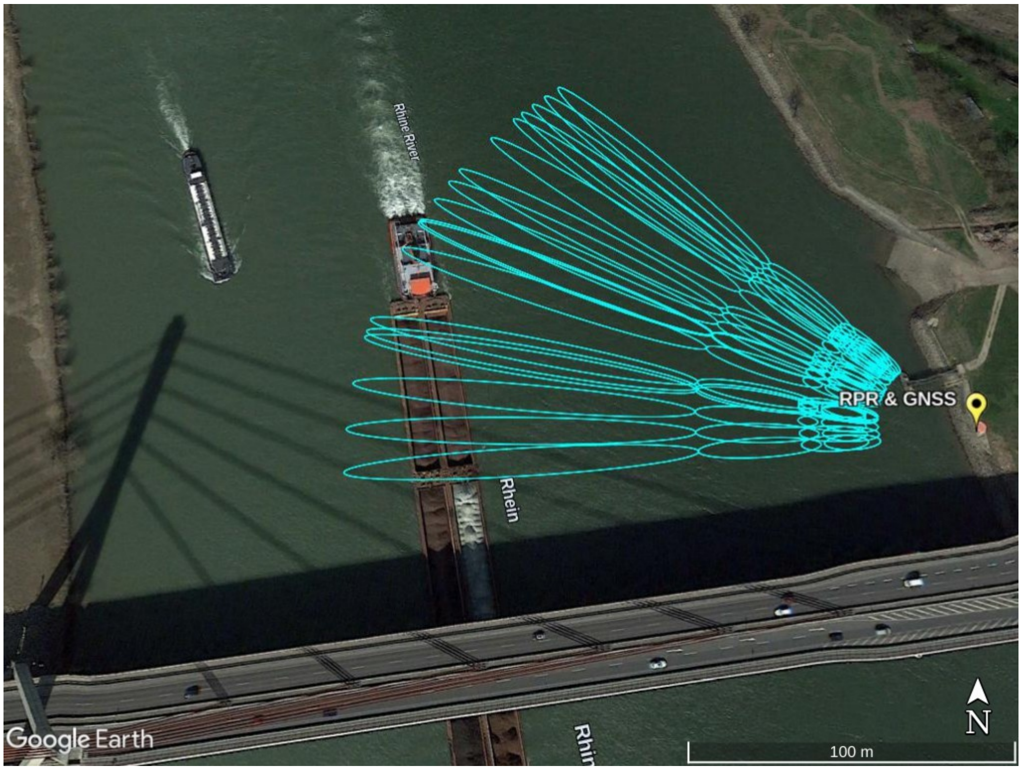

Figure S2. Footprints of the reflected GPS signals and shipping traffic in the Rhine river.

Text S1: Installation guide for RPR

1. Raspberry Pi operating system installation

Raspberry Pi does not have internal disk and built-in Operating System (OS). To set up Raspberry Pi, an operating system needs to be installed onto an SD card. OS Image file is available at the official Raspberry Pi website. Installation instruction can be found here: https://www.raspberrypi.com/software/

Once the Raspberry Pi boots, open Raspberry Pi Configuration under Applications Menu > Preferences > Raspberry Pi Configuration. In the "Interfaces" tab, set SSH, VNC, Serial Port, Serial Console to enabled. Secure Shell (SSH) is a commonly used protocol for providing a secure channel between two computers. We can use SSH to connect from a Linux computer or from the Mac terminal to the Raspberry Pi, without installing additional software. The Virtual Network Connection (VNC) server is enabled for viewing GUI desktop of Raspberry Pi remotely via VNC viewer software installed on a server computer. The GPS module sends and receives NMEA data via the serial communication port to the Raspberry Pi.

After setting up the Raspberry Pi, make sure that it has access to the internet.

2. Installing Arduino integrated development environment (IDE)

2.1 Download the latest version of Arduino IDE (1.8.16) from Arduino website. Note that the processor of Raspberry Pi 3 (4) is a 32- (64)- bit, 700 MHz system on a chip, which is built on the ARM11 architecture, thus we recommend installing "Linux ARM 32 bits" download option: https://www.arduino.cc/en/software

2.2 Uncompress downloaded file:

```
tar -xf arduino-1.8.16-linuxarm.tar.xz
```

2.3 Change your current directory to the created folder and install the Arduino IDE using:

```
cd  arduino-1.8.16
sudo ./install.sh
```

3. Assembling GPS module (MPHW) and connecting to the Raspberry Pi:

General instructions for soldering Adafruit GPS FeatherWing, Adafruit Feather Adalogger microcontroller and stackable header can be found here (follow Section 2.1 in "Tutorial Adafruit GNSS-R.pdf"):

https://github.com/fgnievinski/mphw/blob/master/docs/Tutorial%20Adafruit%20GNSS-R.pdf

Plug the assembled GPS (MPHW) module into Raspberry Pi's USB socket using a USB to Micro USB cable.

4. Downloading Arduino and python codes:

4.1 Change your current directory to:

```
cd /home/pi
```

4.2 Download Arduino and python codes from GitHub using `git` commad:

```
git clone https://github.com/MakanAKaregar/RPR.git
```

A new directory (RPR) is created under `/home/pi`

4.3 Create a new directory for archiving RPR's daily NMEA data:

```
mkdir /home/pi/RPR/data
```

5. Adding and installing Adafruit board support package for Arduino IDE

To enable the Arduino IDE to communicate with Adafruit Feather 32u4 Adalogger board we will need to install a board support package that includes Adafruit FeatherWing driver that allows us to upload Arduino sketch via the Arduino IDE.

5.1 Start the IDE and navigate to File > Preference. Under "Settings" tab add the following URL in "Additional Boards Manager URLs"

https://adafruit.github.io/arduino-board-index/package_adafruit_index.json

5.2 Install Adafruit FeatherWing board

Tools > Board > Boards Manager > Contributed. Select "Adafruit AVR Boards" and click "Install" button.

5.3 Quit and reopen the Arduino IDE. Under Tools > Board, select "Adafruit Feather 32u4"

5.4 Under Tools > Port, select "/dev/ttyACM0"

5.5 Compile and upload the Arduino sketch:

- Open `MPHW.ino` sketch available from */home/pi/RPR/ideCodes/* in the Arduino IDE: File > Open
- ensure that board and port are correct
- compile and upload the `MPHW.ino` to the Adafruit Feather 32u4 board under Sketch > Upload

5.6 Test GPS NMEA data streaming/transmitting to the serial port on terminal

Put the GPS antenna near or outside a window to get satellite signals. You can test if RPR setup is done correctly by the following command:

```
cat /dev/ttyACM0
```

This should give you results resembling the following outputs in your terminal:

```
$GPGGA,171540.000,5043.6373,N,00705.2537,E,1,09,0.85,63.6,M,47.7,M,,*5B
$GPGSV,4,1,13,24,77,293,26,19,41,093,39,15,41,182,15,12,38,224,29*70
$GPGSV,4,2,13,17,34,059,41,37,30,161,,13,23,151,29,10,23,295,16*70
$GPGSV,4,3,13,23,20,259,22,14,11,055,32,25,07,233,18,01,05,020,27*7D
$GPGSV,4,4,13,32,05,320,18*47
$GPRMC,171540.000,A,5043.6373,N,00705.2537,E,0.04,2.26,151121,,,A*6F
$GPRMC,171540.000,A,5043.6373,N,00705.2537,E,0.04,2.26,151121,,,A*6F
211115.log

$GPGGA,171541.000,5043.6373,N,00705.2540,E,1,09,0.88,63.6,M,47.7,M,,*57
$GPGSV,4,1,13,24,77,293,26,19,41,093,39,15,41,182,16,12,38,224,29*73
$GPGSV,4,2,13,17,34,059,41,37,30,161,,13,23,151,28,10,23,295,16*71
$GPGSV,4,3,13,23,20,259,22,14,11,055,32,25,07,233,18,01,05,020,27*7D
$GPGSV,4,4,13,32,05,320,18*47
$GPRMC,171541.000,A,5043.6373,N,00705.2540,E,0.02,25.95,151121,,,A*55
$GPRMC,171541.000,A,5043.6373,N,00705.2540,E,0.02,25.95,151121,,,A*55
211115.log
```

6. Updating GPS module’s firmware

To generate the SNR data with 0.1-dB precision we should update Adafruit GPS FeatherWing’s firmware. To perform the firmware update, we require a Windows computer to install The GlobalTop Flash Tool software which allows updating the firmware of GPS chip. The custom firmware for the MediaTek GPS can be made available upon request. After updating the GPS firmware, plug the GPS module into the Raspberry Pi and redo steps from 5.3 to 5.6.

You can test if the GPS firmware is upgraded by the following command:

```
cat /dev/ttyACM0
```

This should give you the following sentences in your terminal:

```
$GPGGA,172644.000,5043.6391,N,00705.2406,E,1,5,1.50,66.1,M,47.7,M,,*67
$GPGSV,2,1,06,24,82,295,26.2,12,44,227,32.4,19,42,087,44.2,17,32,054,42.
4*7E
$GPGSV,2,2,06,13,18,152,34.6,23,17,255,24.6*76
$GPRMC,172644.000,A,5043.6391,N,00705.2406,E,0.14,290.38,151121,,,A*63
$GPRMC,172644.000,A,5043.6391,N,00705.2406,E,0.14,290.38,151121,,,A*63
211115.log

$GPGGA,172645.000,5043.6391,N,00705.2406,E,1,5,1.50,65.9,M,47.7,M,,*6D
$GPGSV,2,1,06,24,82,295,25.0,12,44,227,32.3,19,42,087,44.4,17,32,054,42.
0*7A
$GPGSV,2,2,06,13,18,152,34.4,23,17,255,25.2*71
$GPRMC,172645.000,A,5043.6391,N,00705.2406,E,0.14,290.38,151121,,,A*62
$GPRMC,172645.000,A,5043.6391,N,00705.2406,E,0.14,290.38,151121,,,A*62
211115.log
```

7. Installing some python packages:

Make sure your Raspberry Pi is connected to the internet. We will need a few additional python packages to make the python scripts work. Running the following command from terminal will install these libraries:

6.1 `sudo apt-get update`
6.2 `sudo pip3 install pyserial`
6.3 `sudo pip3 install timezonefinder`
6.4 `sudo pip3 install pytz`
6.5 `sudo pip3 install pynmea2`

8. Setting crontab jobs

We automate data picking and Raspberry Pi's clock synchronization by setting up two boot-based cron jobs that run python codes whenever the Raspberry Pi boots up. To create a crontab file, execute the following commands in a terminal:

8.1 *`crontab -e`*

You can select a text editor to make changes to the crontab file. Select your desired text editor (e.g. nano, vi ...).

8.2 Add these lines to the crontab file:

```
#to parse RPR nmea data into daily files
@reboot sudo /bin/python3.7 /home/pi/RPR/pyCodes/dataPicker.py

#to sync Raspberry Pi's clock with GPS time (it is local time)
@reboot sudo /bin/ python3.7 /home/pi/RPR/pyCodes/setPiClock.py
```

Note that for Raspberry PI 3 B and B+ the python source code is at `/bin/python3.7` and for Raspberry PI 4 at `/usr/bin/python3.7`

A simple code can be added to the time-based cron job scheduler for compressing daily NMEA files and transfer to a server. For example, the python code `gzipForCron.py` will be run every day at 1:00 AM.

```
#to compress daily RPR files and transfer to a remote server
00 01 * * * /bin/python3.7 /home/pi/RPR/pyCodes/gzipForCron.py
```

Ensure that you included the correct file path in your crontab command.