MDPI

Article

Design of a MATLAB HEC-RAS Interface to Test Advanced Control Strategies on Water Systems

Ronan Deshays, Pablo Segovia † and Eric Duviella *

CERI Systèmes Numériques, IMT Lille Douai, Univ. Lille, F-59000 Lille, France; ronan.deshays@etu.imt-lille-douai.fr (R.D.); P.SegoviaCastillo@tudelft.nl (P.S.); eric.duviella@imt-lille-douai.fr (E.D.)

† Current address: Maritime and Transport Technology, Faculty of Mechanical, Maritime and Materials Engineering, Delft University of Technology, 2628CD Delft, The Netherlands.

Abstract: The software package HEC-RAS (Hydrologic Engineering Center's River Analysis System) is widely used by the water engineering community to analyze hydraulic systems and perform development planning. Furthermore, it integrates a control module that allows implementing basic controllers. For more complex approaches, developers from the automatic control and artificial intelligence (AI) communities usually design, implement, and test new algorithms using dedicated software such as MATLAB. However, models of hydraulic systems employed in MATLAB are often very simple. The main objective of the paper is to design a simulation architecture by coupling HEC-RAS with MATLAB, thus improving the accuracy of the dynamics of the hydraulic systems considered in the control simulations. The main feature of the MATLAB HEC-RAS interface design is that it allows one to execute customized code at regular time intervals during the simulation. In this way, closed-loop control and optimization algorithms can be implemented and tested. Moreover, the generic interface allows for any configuration of hydrographical systems. The proposed interface is presented in this paper, and the performance of the approach is demonstrated considering two case studies of different nature.

Keywords: HEC-RAS; MATLAB; water systems; simulation; model integration; control design; automation

Citation: Deshays, R.; Segovia, P.; Duviella, E. Design of a MATLAB HEC-RAS Interface to Test Advanced Control Strategies on Water Systems. *Journal Not Specified* **2021**, *1*, 0. https://doi.org/

Received:
Accepted:
Published:

1. Introduction

Hydrographical networks are large-scale systems and consist of natural rivers and artificial canals. Their management, which aims to meet human needs, e.g., avoid natural disasters such as floods, can be divided into two categories: structural and non-structural. On the one hand, structural management aims at defining the infrastructure facilities that allow to fulfill the defined objectives. It is of long-term nature, requires the definition and execution of development planning, and responds to environmental guidelines elaborated with all stakeholders in mind. Managers are usually assisted by engineers and advisors that employ hydraulic software tools, e.g., Hydra (http://hydra-software.net/), Mike11 (https://www.mikepoweredbydhi.com/products/mike-11), SWMM (https://www.epa.gov/water-research/storm-water-management-model-swmm) and HEC-RAS (https://www.hec.usace.army.mil/software/hec-ras/). These software packages allow one to carry out 2D/3D modeling of rivers and canals as well as hydraulic devices and infrastructures, and are used to simulate scenarios and design equipment. On the other hand, the main purpose of non-structural management is that of guaranteeing an efficient management of these systems without modifying the existing structure and equipment. In this case, adaptive and predictive management strategies are required, and they can benefit from forecasts of meteorological events and human activities.

A literature survey reveals that the design of advanced control strategies for water systems has been and still is an active research topic, as attested by the large number

of scientific publications in the last years [1–12]. However, these approaches do not seem to have found much success in practice yet. Indeed, system managers often opt to implement manual control strategies, in which human operators adjust the actuators, e.g., gate openings, according to their own judgment. Naturally, this course of action often results in far-from-optimal system performances [13]. Another approach consists in governing the systems using expert rules, i.e., a set of simple logic rules based on historical management guidelines and the expertise of operators [14–17].

Although the approaches used by both groups are rather different in nature, the main objectives are aligned. Indeed, the conception of a management strategy that ensures the optimization of water resources is of the utmost importance, especially in the current climate change context. Therefore, it seems logical to focus on the development of a methodology that is suitable for the needs of both communities and which can facilitate their interaction. In this regard, it might be of interest to develop an interface that allows combining two software tools of different nature. On the one hand, numerical computing environments with the capability to design advanced control strategies, e.g., MATLAB, and on the other hand, dedicated hydraulic software packages that can provide accurate representations of the dynamics of hydrographical networks, e.g., HEC-RAS.

Research on the potential benefits of an interface between MATLAB and HEC-RAS has not attracted a lot of attention except for [18]. The authors provide HEC-RAS specifications together with several useful scripts for basic operations, which are illustrated using a complete example. Their interface allows MATLAB to control HEC-RAS by considering unique sequences, i.e., hydrographs over long periods of time. Therefore, Monte Carlo simulations can be performed using HEC-RAS and MATLAB [19, 20]. However, control algorithms that require a feedback loop cannot be implemented using their approach, as there exists no interface that enables it.

Building on the existing results, this work tackles the design of a MATLAB HEC-RAS interface that provides both researchers and practitioners with a tool to test MATLAB control algorithms using the capabilities that HEC-RAS offers in terms of model accuracy. The developed interface is a ready-to-use solution that only requires using the MATLAB command window and editing some m-files to automate HEC-RAS computations. Moreover, it deals with any HEC-RAS project-related task, including writing, reading, and extracting results, thus allowing the user to focus on the control algorithm and the results. Indeed, a successful initialization of the interface, which can be achieved by simply supplying a stable HEC-RAS project to the interface, guarantees that the user will never need to use the HEC-RAS GUI (Graphical User Interface). Furthermore, the proposed solution allows to perform closed-loop simulations, thus extending the results reported in [18]. Last but not least, the development of such interface allows the automatic control and artificial intelligence (AI) communities to benefit from the expertise of the hydraulic and environmental engineering communities. Indeed, the knowledge about detailed hydraulic and environmental phenomena from the former is rather limited compared to that of the latter. This tool should also be interesting to hydraulic and environmental researchers, allowing them to develop more faithful models from which automatic control and AI researchers would definitely benefit, and would allow them to design innovative approaches.

Note that the main contribution with regard to the state-of-the-art is the set of codes employed by the interface to allow for the exchange of information between MATLAB and HEC-RAS. As stated before, these are offered as a ready-to-use solution and are accompanied by exhaustive supporting documentation that provide insight on the potential of the solution. Information on how to access codes and documentation is provided in the *Data Availability Statement* located at the end of the paper. Furthermore, the present paper describes the main features of the interface and presents some examples.

The remainder of this paper is structured as follows. Section 2 presents some updates on the HEC-RAS controller discussed in [18] and provides an overview on the

vast diversity of HEC-RAS versions. The MATLAB HEC-RAS interface is detailed in Section 3. Section 4 illustrates the approach by considering two case studies (a canal and a river) and three scenarios, which allows to demonstrate the effectiveness of the proposed approach.

2. The HEC-RAS Controller

2.1. Description

The HEC-RAS software package is developed and maintained by the Hydrologic Engineering Center (HEC) of the US Army, and further support is provided by means of a documentation website [21] and a discussion forum (https://www.kleinschmidtgroup.com/the-ras-solution). An HEC-RAS Application Programming Interface (API) simplifies the one-dimensional (1D) water system project development task and has three classes: HEC-RAS geometry, HEC-RAS flow, and HEC-RAS controller.

- HEC-RAS geometry allows creating models of hydrographical systems based on their geometrical features and other physical parameters, e.g., roughness coefficient.
- HEC-RAS flow aims at reproducing the dynamics of water systems in steady and unsteady flows using specific information, e.g., boundary and initial conditions and hydrographs.
- HEC-RAS controller allows to automate tasks such as running projects, data retrieval, postprocessing output data, and, in a future release, computational tools such as Monte Carlo Simulations.

However, there exists no official supporting documentation for the HEC-RAS controller, being [22] the most complete document regarding such class. The proposed procedure consists in coding the HEC-RAS controller using VBA (Visual Basic for Applications) and automating HEC-RAS using Excel. HEC-RAS provides objects, e.g, classes, functions, and methods, that can be used by many object-oriented programming languages, enabling their interfacing. This allows the developer to govern HEC-RAS from an external source, using the programming language of choice.

Although these functions work well and are reliable, the possibilities offered by this approach may be somewhat limited. For instance, there is no available method to edit boundary conditions directly from the code when this object is used in MATLAB. However, the structure of the HEC-RAS files allows to overcome this issue. Even though the HEC-RAS controller deals mainly with binary files, HEC-RAS projects can always be edited using plain text files. Indeed, text files are easier to manage and deal with than binary files. Moreover, the parameters that the user might be interested in modifying are usually stored in plain text files. Consequently, reading and editing HEC-RAS projects from MATLAB can be performed in a straightforward manner.

Table 1 presents a list of important HEC-RAS files, where X can be any digit. The reader is referred to the HEC-RAS file types post [23] and the HEC-RAS documentation [21] for a detailed description of the structure of HEC-RAS files.

Table 1. List of important HEC-RAS files.

File Extension	File Type	Description
.pXX	Plain text	Plan
.uXX	Plain text	Unsteady flow, boundary and initial conditions
.dss	Binary	Temporarily stores HEC-RAS computation results, can be read using HEC-RAS controller
.rst	Binary	Restart file, contains information to be used at the start of the next HEC-RAS step

2.2. HEC-RAS API Stability

The HEC-RAS controller, as part of the HEC-RAS API, is subject to changes with every release of a new HEC-RAS version. Moreover, new MATLAB releases come

out periodically, a fact that causes some functions in [22] not to be available in certain MATLAB releases. This issue might lead to errors during the interfacing of HEC-RAS and MATLAB. If such an error arises, the MATLAB function `methodsview` yields a list of HEC-RAS controller methods supported by MATLAB, which can be used to check whether a certain MATLAB version is compatible with an HEC-RAS controller version. The same function displays information about variables that need to be adapted in MATLAB.

As a side note, the code given in [18] to control HEC-RAS with MATLAB R2015b [24] has an upward compatibility with more recent MATLAB releases. In fact, the interface was tested with MATLAB R2020a [25]. The reason why this script supports a change of MATLAB version is that no advanced MATLAB functions are used. Indeed, the script mainly performs file reading and writing operations as well as some basic mathematical functions. However, this may not be true for parallel computing.

3. Proposed MATLAB HEC-RAS Interface

3.1. Interface Design

The main goal of the proposed interface is to allow the performance of step-by-step simulations. Indeed, while HEC-RAS offers an accurate simulation of the dynamics of water systems, MATLAB enables implementing advanced control algorithms. Thus, the performance of closed-loop control algorithms can be tested on a simulator while considering the real dynamics. It is therefore necessary that HEC-RAS sends the values of certain variables, e.g., water levels, at regular intervals to MATLAB. Then, based on this information, the control algorithm can determine the suitable actions to be applied next by the available hydraulic devices, e.g., gates and pumps, in HEC-RAS, taking into account the control frequency. Both software tools must run in parallel, exchanging information at fixed time intervals.

The design of the MATLAB HEC-RAS interface is directly inspired from the work in [26]. A MATLAB transcription of the offtake flow example in [22] is then performed. In doing so, a modified version of the offtake flow example is supported by the MATLAB interface. Note that the HEC-RAS objects that already integrate automation capabilities, e.g., navigation dams, are not supported yet by the interface.

3.2. Overall Functioning

The MATLAB HEC-RAS interface performs a step-by-step simulation, dividing a long simulation into a set of smaller ones, which will be computed one at a time, one after the other. Figure 1 presents a simplified diagram of the interface operation. Note that a model of the water system is required by the HEC-RAS geometry and flow classes, which are encapsulated in the HEC-RAS process block. The directory tree of files that constitute the model is called the HEC-RAS project.

The file `settings.ini` defines the HEC-RAS and MATLAB project path; the start and end time of the simulation; the points of interest in the water systems, e.g., location of sensors and actuators; and the type of data which will be saved in the results array, e.g., flow and level.

When the interface is launched, the master function `master.m` instantiates several variables. A part of this function, which is dedicated to the preparation of the first computation step in HEC-RAS, is presented in Listing 1. Note that the management policy for the rest of iterations is not included. The HEC-RAS controller manages the ON/OFF switch for the HEC-RAS process. Moreover, the latter requires information contained in the HEC-RAS text files. Text file functions in MATLAB are dedicated to edit and set HEC-RAS text files, guaranteeing the conditions for the first simulation step.

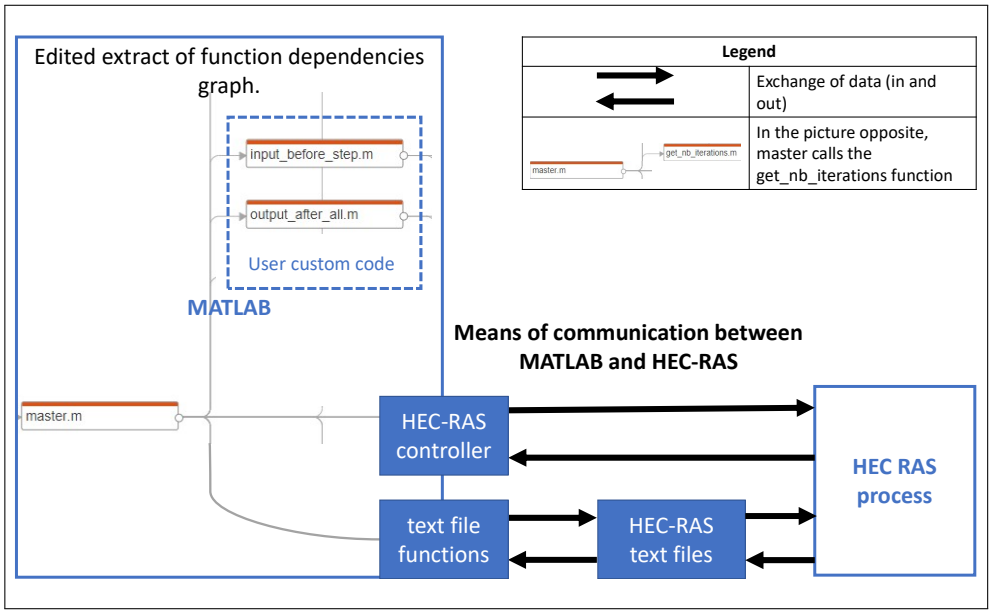

Figure 1. Diagram of the MATLAB HEC-RAS interface operation.

```
    for iteration_nb=1:nb_of_iterations
        if iteration_nb==1 % RAS initialization (no
     restart)
            % Launch custom input scripts before
    RAS initialization
            input_init(files, sett.XSlist)
            % Do not use restart file for the first
     step
                update_param_first (files.flow, '
    Use Restart', 'Use Restart=0')
                [line_nb,found] = find_str_after(
    files.flow, 0, 'Restart Filename');
                if found
                    param_delete (files.flow,
    line_nb)
                end
            % Set simulation date for the first
    step
                Temp2 = First.increment(time_step);
                temp = ['Simulation Date=',First.
    HRset,',', Temp2.HRset]; % One RASnewline is
    already in update param function
                update_param_first (files.plan,'
    Simulation Date', temp)
            % Write restart file at the end of the
    simulation
                update_param_first (files.plan, '
    Write IC File at Sim End', 'Write IC File at
    Sim End=-1')
        else % Different management for the rest of
     iterations (not presented)
        end
```

Listing 1: Code snippet of `master.m`.

After the initialization, the simulation starts with the first computation. The HEC-RAS controller informs the interface once the computation has been performed. The results can be requested from the HEC-RAS controller and exported into a MATLAB cell array. Prior to the execution of the second step, the interface edits the HEC-RAS text files to shift the simulation date forward in time, and eventually apply a control action, if required by the user custom code. Consider again Figure 1, which features two of the four scripts that can contain user custom code. The script `input_before_step` is

called before each simulation step, and can be used to modify gate openings according to the water stage elevation (WSE) value provided by the script `output_after_step` (not included in Figure 1). Then, a change can be made to the current gate opening by editing the corresponding value in the HEC-RAS files, which will be taken into account at the next step. The simulation is resumed after the modifications have been completed. This iterative process continues until the end date specified by the user (in hours, minutes, and seconds) in `settings.ini`. Note that the sampling time is also selected by the user.

Thus, each simulation step starts with the results of the previous step as the initial condition. To illustrate this simulation process, an example is depicted in Figure 2, where a master script is executed during the entire simulation. The HEC-RAS process is paused at the end of each step and restarted at the beginning of the next. The user custom scripts are called regularly, with the exception of `input_init` and `output_after_all`, which are called only once. More precisely, `input_init` is called after the first start of an HEC-RAS process and before any HEC-RAS computation. Its purpose is that of specifying the boundary and initial conditions, or to reset any hydrograph. On the other hand, `output_after_all` is called once the simulation has been completed, and is employed to analyze the simulation results. Furthermore, the files `input_before_step` and `output_after_step` are run throughout the simulation, between two consecutive simulation steps. They are used to save the results obtained in the current step and update the initial conditions for the next step. Note that this philosophy is particularly well suited for control algorithms that are implemented in MATLAB.

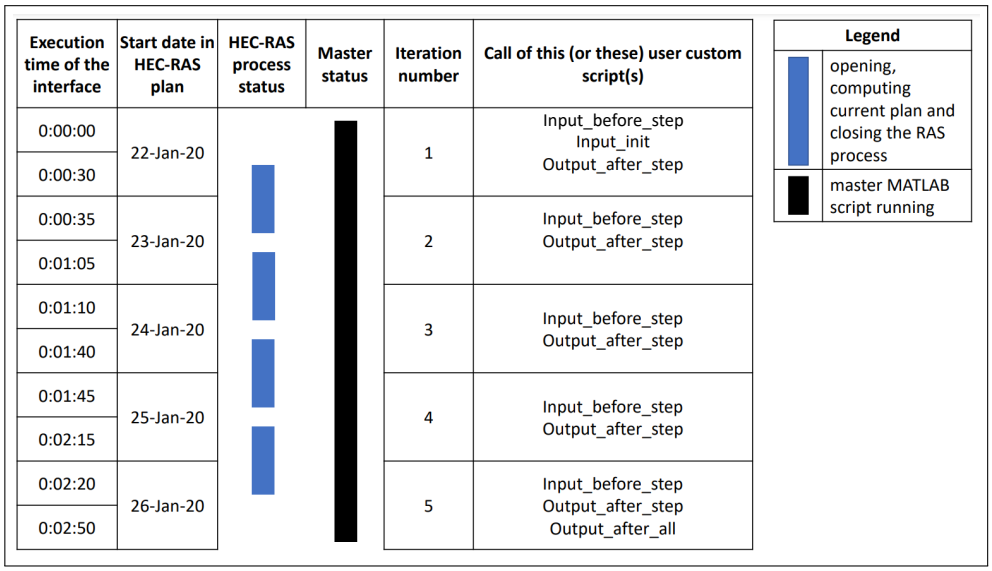

Figure 2. Illustration of the MATLAB HEC-RAS interface operation allowing step-by-step simulations.

Technical information about user custom scripts and description of supported parameters are available in the MATLAB HEC-RAS interface documentation [27].

3.3. Some Specifications of the Proposed Approach

The interface consists in several MATLAB functions that are sorted by type in a directory tree, presented in Figure 3. Relevant documentation is provided so that each function can be understood and called directly without using the master script. Moreover, Table 2 provides a short description of each important folder in the directory tree. Note that the MATLAB scripts in the parent folder have already been presented in the previous section, with the exception of `installer.m` in the installation folder. This script is designed to allow for the installation of multiple interface versions and verify that every required file is present.

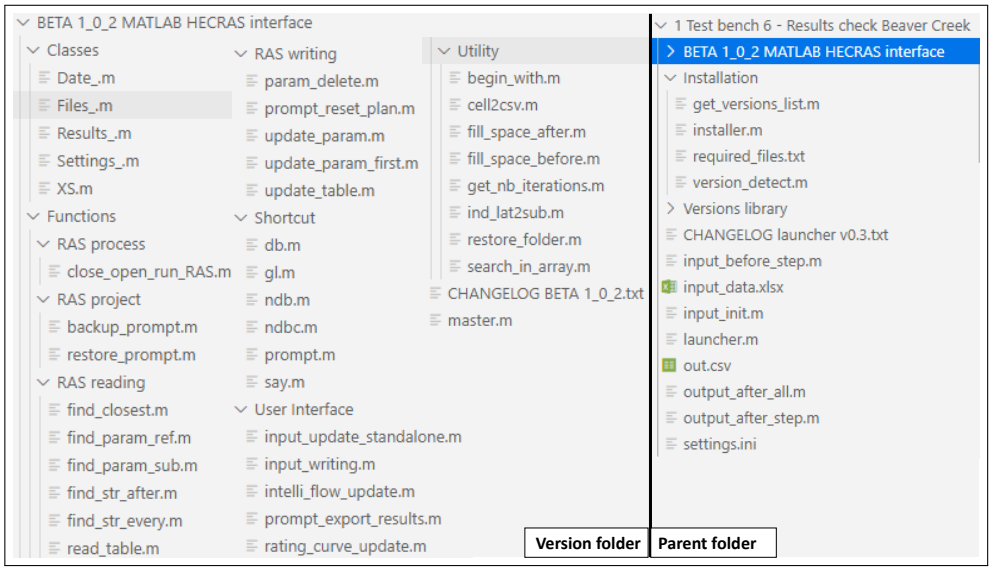

Figure 3. Screenshots of the directory tree of the MATLAB HEC-RAS interface code.

Table 2. List of important folders.

Name	Description
Classes	Classes for object-oriented programming, used as enhanced structures. Variables of these classes store information in a format compatible with HEC-RAS (e.g., XS) or information given by the user (e.g., Settings_)
RAS reading	Low-level functions, which look for parameters in a given HEC-RAS form and return position line number(s)
RAS writing	Low-level functions that replace a given line in an HEC-RAS file with a given string
Shortcut	Simple functions (unrelated to HEC-RAS) to help developers save time
User interface	High-level functions that use RAS reading and writing in a special context to make the interface smarter and help developers save time
Utility	Low-level functions used to avoid code repetition for specific tasks

However, to avoid further coding work from the user, the interface has been coded to be used directly. Indeed, it is only necessary to specify the HEC-RAS data that are required by the control algorithms, as well as the control setpoints that must be sent to the hydraulic devices in HEC-RAS. Therefore, it is recommended to use the master script as a pattern, to identify which functions must be called, and the execution order. The reader is referred to the interface documentation [27] for detailed information on possible workarounds using the interface. Moreover, further insight on how to edit the interface is also provided.

4. Case Study

The MATLAB HEC-RAS interface is tested first on an academical example, a rectangular canal, and then on a model of Beaver Creek to set up a control rule in the spirit of the work in [22]. The first goal is to show that the automated start and stop simulations performed thanks to the designed interface lead to the same results as a manual simulation without restart. The second objective is that of defining closed-loop control

rules for gates using the automated start and stop simulation to validate the approach. Moreover, the one-step simulations are performed by assuming that all hydrographs and control operations of gates are known beforehand for the whole simulation. Furthermore, note that no sudden modification of the input/output flows occurs during the one-step simulations. Likewise, the gate opening values do not change abruptly between two consecutive sampling instants. Indeed, HEC-RAS performs linear interpolation between the two values. Consequently, the resulting control strategies are faithful to reality, and are implemented in the custom user scripts.

4.1. First Case Study: The Rectangular Canal

The considered canal is 20,000 m long, with a width of 20 m and a normal depth of 9 m, while Manning's roughness coefficient is equal to 0.04 for the whole canal. A cross section XS is considered every 10 m, except for the last one, and the geometry of all cross sections are exactly the same. This information is summarized in Figure 4. There is one measurement station located at XS 700. Moreover, it is possible to simulate an offtake flow at XS 500. The input flow of the canal is controlled upstream, at XS 1000.

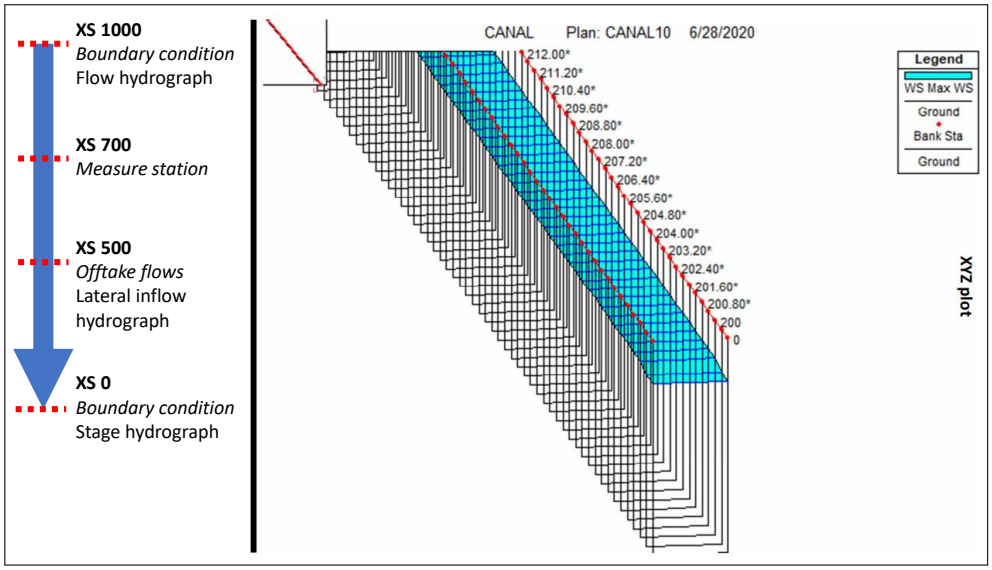

Figure 4. Schematic representation of the geometry of a simple rectangular canal.

The evolution of the system is simulated for ten hours, with a control sampling time of one hour. The rationale behind this choice is motivated by the fact that the available data were sampled every thirty minutes, with some missing data. Therefore, re-sampling these data every hour allows to bypass this issue and fill in the missing data with average values. Nevertheless, the interface allows to consider the desired sampling time. Moreover, and while MATLAB scripts are executed every hour, the computation interval in HEC-RAS is equal to ten minutes. This means that the HEC-RAS process is executed six times per one MATLAB execution.

The initial condition considered downstream, i.e., XS 0, is given in Table 3 as a stage hydrograph. The simulated upstream hydrographs, i.e., XS 1000, are defined according to two scenarios. The first one, labeled as *scenario1*, is defined based on the pattern given in Table 3. At the initial time, the upstream flow is equal to zero during ten minutes, then it is equal to 10 m^3/s for the next ten minutes, and so on. The second scenario, labeled as *scenario2*, corresponds to a constant upstream inflow of 25 m^3/s, which can only cause the water level to increase. To mitigate this, the offtake flow located at XS 500 is controlled to withdraw water from the canal, thus avoiding overflow. The control rule described by Algorithm 1 consists in withdrawing water from the canal when the WSE at XS 700 exceeds 9.3 m. Note that a negative flow implies that HEC-RAS diminishes the canal flow, following the sign convention in [22].

Table 3. Boundary conditions of the rectangular canal.

Scenarios	Rule State	Name	XS	Pattern Values
1	No rule	Flow hydrograph	1000	{0, 10, 20, 30, 20, 10} m^3/s
2	With rule	Flow hydrograph	1000	Constant, 25 m^3/s
Both	-	Stage hydrograph	0	9 m

Algorithm 1 Control rule for the rectangular canal

if WSE at XS 700 is higher than 9.3 **then**
  set next value of lateral inflow hydrograph at XS 500 equal to -100 m^3/s
else
  set next value of lateral inflow hydrograph at XS 500 equal to 0
end if

In this first case study, and for both scenarios, the user must define the `settings.ini` file as indicated in Listing 2. Note that the starred values for XS 2, XS 3, and XS 4 denote interpolated cross sections.

```
StartTime = 22-Jan-2020 00:00
EndTime = 22-Jan-2020 10:00
TimeStep = 00:01:00:00

XS1 = CANAL,CANAL,1000
XS2 = CANAL,CANAL,700.00*
XS3 = CANAL,CANAL,500.00*
XS4 = CANAL,CANAL,300.00*
XS5 = CANAL,CANAL,0

ProjectPath = F:\CanalModif2\
ProjectName = CANAL
```

Listing 2: Definition of `settings.ini` for the rectangular canal.

Figure 5 shows the water level at XS 700 for *scenario1*. The level increases and decreases according to the predefined upstream flow. The HEC-RAS results with automated start and stop simulations are depicted in dashed red, while the results without restart correspond to the continuous blue line. Note that the results obtained are identical for both methods, confirming that the MATLAB HEC-RAS interface approach performs as desired.

Regarding the second scenario, i.e., *scenario2*, the gate control rule is implemented in `input_before_step.m` as shown in Listing 3. This script is executed every hour, in accordance with the defined sampling time. Note that the variable *ws_elev* is assigned the current WSE (read from HEC-RAS), and *xs{3}* corresponds to an offtake flow, which is applied at the next steps of HEC-RAS.

On the other hand, Listing 4 shows how to access the WSE information from HEC-RAS using `output_after_step.m`.

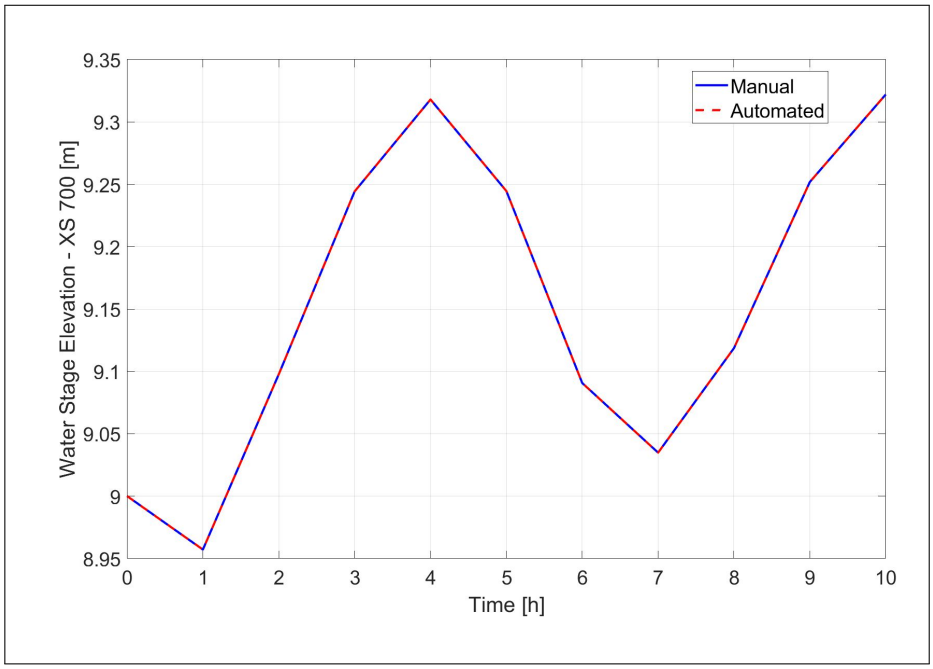

Figure 5. WSE at XS 700 for the rectangular canal—*scenario1*.

```
function input_before_step (iteration_nb, xs, files
    )
    global ws_elev
    global data
    if iteration_nb > 1
        if ws_elev > 9.3
            data(iteration_nb+1) = -100; % m^3/s
        end
    else
        data = zeros(1,100);
    end
    input_list = {data, 'hydrograph', '.u', '
    Lateral Inflow Hydrograph', xs{3}};
```

Listing 3: Definition of `input_before_step.m` for the rectangular canal.

```
function output_after_step (results, xs, rp)
    global ws_elev
    ws_elev = results{2}.GetValue(xs{2}, rp, 'W.S.
    Elev', 'last'); % read WSE at XS 700.00*
```

Listing 4: Definition of `output_after_step.m` the for rectangular canal.

Figure 6a depicts the level at XS 700 for *scenario2*, when the control rule is implemented. This rule requires to withdraw water from the canal whenever the level exceeds a height of 9.3 m. The lateral inflow hydrograph applied by the control rule to the offtake gate at XS 500 is presented in Figure 6b. Note again that both approaches yield the same results. Thus, *scenario2* demonstrates the effectiveness of the control approach defined in MATLAB and its application in HEC-RAS thanks to the interface.

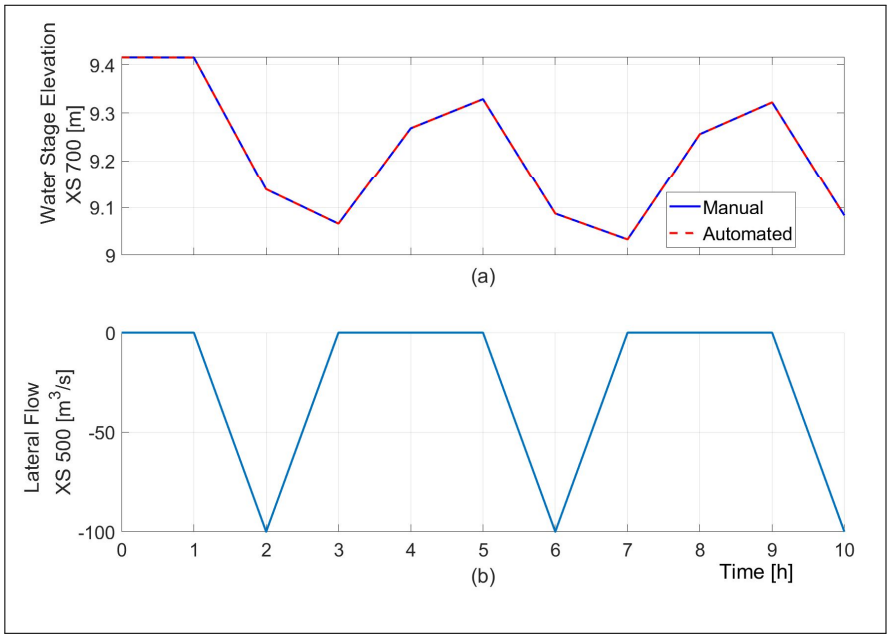

Figure 6. Rectangular canal—*scenario2*. (**a**) WSE at XS 700. (**b**) Lateral inflow hydrograph applied at XS 500.

The most interesting feature of the first case study is that it allows to demonstrate the coordination possibilities between MATLAB and HEC-RAS. Moreover, advanced control algorithms can be easily implemented instead of the simple control rules, which only requires replacing the latter with the algorithm of choice. Indeed, this first case study depicts the basic functioning of the interface. In contrast, the following case study is closer to a real scenario.

4.2. Second Case Study: Beaver Creek

The Beaver Creek case study was proposed in [22] (p. 116), and is based on the Bridge Hydraulics project, available for download on HEC-RAS website [21]. Further information on how to set up and run this case study can be found on the MATLAB HEC-RAS interface documentation ([27], Tutorial 1: simple automation example), which provides a step-by-step explanation to build a simplified version of this project.

Figure 7 presents the Beaver Creek HEC-RAS model with the cross sections and the locations of the measurement stations and the two gates, i.e., XS 5.61 and XS 5.44. The offtake gate located at XS 5.61 is used to withdraw water from the canal. Conversely, the inflow gate located at XS 5.44 is used to supply the canal with water. The initial and boundary conditions are summarized in Table 4, and consist of a constant stage downstream hydrograph at XS 5.0 and a modification over time of the upstream flow at XS 5.99. Initially, the upstream flow is equal to zero, then it increases to 50 m^3/s for the next period, and so on. The sampling time for the HEC-RAS project and MATLAB are equal to twelve hours and one day, respectively.

The user must define, in `settings.ini`, the control points XS1 and XS2 that correspond to the cross sections XS 5.61 and XS 5.44, respectively. This is shown in Listing 5.

A new scenario is defined for the Beaver Creek case study, i.e., *scenario3*, which modifies the upstream hydrograph and implements a control rule in MATLAB. The rule controls the two gates based on conditions on the WSE value at XS 5.61. When this value exceeds a height of 65 m, the offtake gate located also at XS 5.61 opens, withdrawing 100 m^3/s. As soon as the WSE is lower than 65 m, this gate is closed. On the other hand, when the WSE is lower than 64.5 m, the inflow gate opens to supply the canal with 20 m^3/s. These control rules are summarized in Algorithm 2. Moreover, the file `input_before_step.m` is used to implement Algorithm 2.

Algorithm 2 Control rule for the Beaver Creek case study

if WSE in XS 5.61 is higher than 65 m **then**
set next value of lateral inflow hydrograph at XS 5.61 equal to -100 m^3/s and XS 5.44 to zero
else if WSE in XS 5.61 is lower than 64.5 m **then**
set next value of lateral inflow hydrograph in XS 5.44 equal to 20 m^3/s and XS 5.61 to zero
else
set next value of lateral inflow hydrograph at both XS 5.61 and XS 5.44 equal to zero
end if

Table 4. Boundary conditions for the Beaver Creek case study.

Name	XS	Pattern Values
Flow hydrograph	5.99	{0, 50, 100, 150, 200, 150, 100, 50} m^3/s
Stage hydrograph	5.0	64.7 m

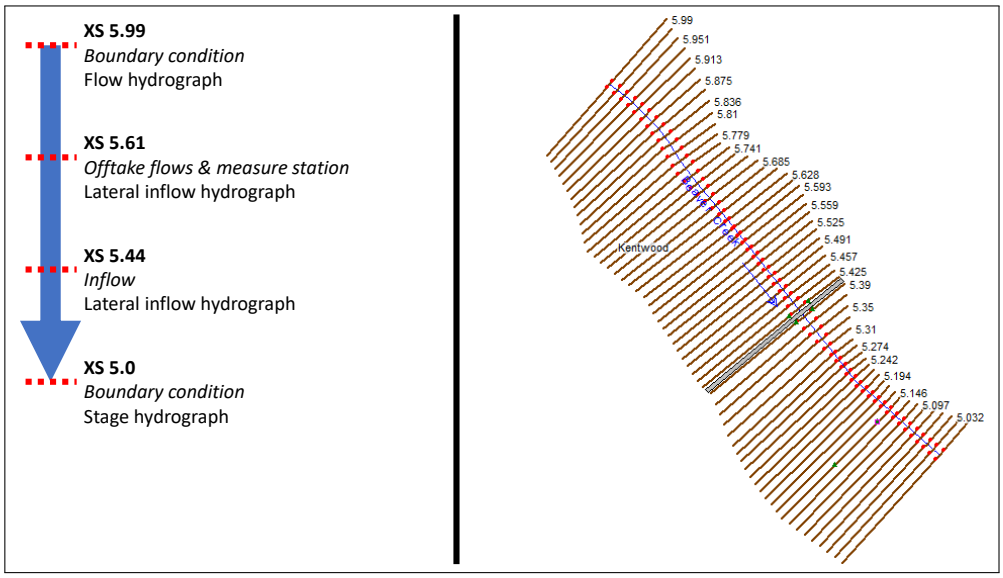

Figure 7. Geometry of the Beaver Creek case study.

```
StartTime = 03-Jan-2014 00:00:00
EndTime = 13-Jan-2014 00:00:00
TimeStep = 01:00:00:00

XS1 = Beaver Creek, Kentwood, 5.61
XS2 = Beaver Creek, Kentwood, 5.44

ProjectPath = F:\Beaver new rule\
ProjectName = beaver
```

Listing 5: Definition of `settings.ini` for the Beaver Creek case study.

```
function input_before_step (iteration_nb, xs, files
    )
    global ws_elev
    global data1 data2
    if iteration_nb > 1
        if ws_elev > 65
            data1(iteration_nb+1) = -100; % m^3/s
            db(data1)
        elseif ws_elev < 64.5
            data2(iteration_nb+1) = 20; % m^3/s
            db(data2)
        end
    else
        data1 = zeros(1,100);
        data2 = zeros(1,100);
    end
    input_list = {data1, 'hydrograph', '.u', '
    Lateral Inflow Hydrograph', xs{1}; data2, '
    hydrograph', '.u', 'Lateral Inflow Hydrograph',
     xs{2}};
```

Listing 6: Definition of `input_before_step.m` for the Beaver Creek case study.

Then, the WSE values can be obtained from HEC-RAS by conveniently modifying the `output_after_step.m` file.

```
function output_after_step (results, xs, rp)
    global ws_elev
    ws_elev = results{1}.GetValue (xs{1},rp, 'W.S.
    Elev','last');
```

Listing 7: Definition of `output_after_step.m` for the Beaver Creek case study.

Figure 8a shows the automated start and stop results in dashed red, and the results without restart using a continuous blue line. Moreover, Figure 8b,c depicts the lateral inflow hydrographs applied at XS 5.61 (offtake gate) and XS 5.44 (inflow gate), respectively.

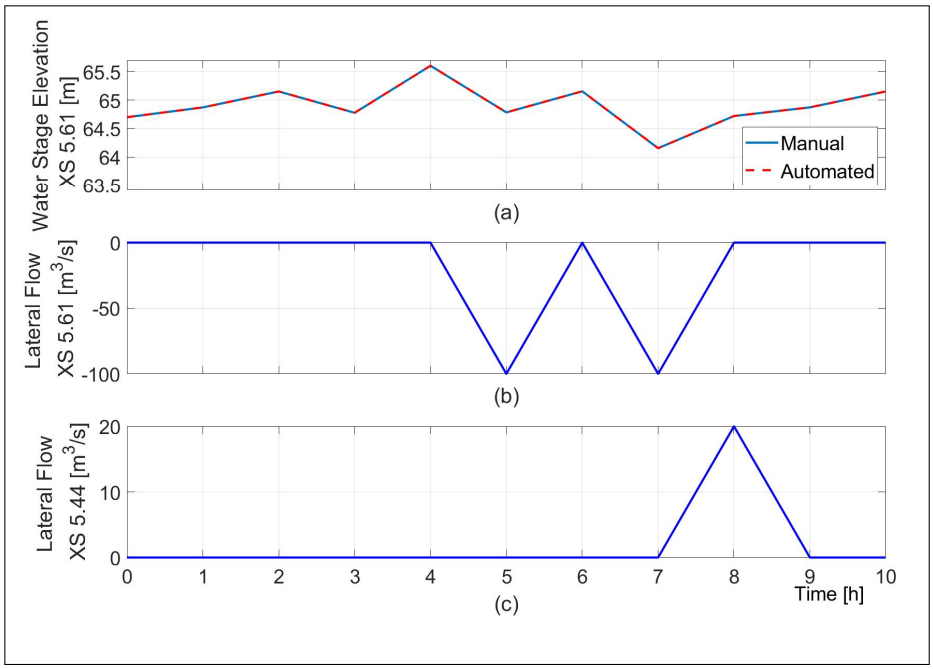

Figure 8. Beaver Creek case study—*scenario3*. (**a**) WSE at XS 5.61. Lateral inflow hydrographs applied at (**b**) XS 5.61 and (**c**) XS 5.44.

scenario3 shows that the upstream flow modification and the control of the several gates based on one or several measurements of WSE are enabled by the MATLAB HEC-RAS interface. This approach can be easily extended to deal with advanced control algorithms for more complex systems, benefiting from the multiple functionalities offered by MATLAB and HEC-RAS.

5. Conclusions

The work performed in this paper aims at contributing to the issue of providing an appropriate interface between HEC-RAS and MATLAB, bearing in mind the first steps carried out in [18]. More precisely, a MATLAB HEC-RAS interface has been designed to allow for closed-loop simulations of hydraulic systems that make use of advanced control algorithms. The proposed solution is coded as a series of simple MATLAB scripts. The main objective is that researchers in the Automatic Control and Artificial Intelligence communities can focus on testing control approaches. At the same time, it also aims at providing hydraulic researchers and engineers with a tool to include MATLAB functionalities in their simulations. In particular, the proposed interface allows one to read and write HEC-RAS files, perform fully automated step-by-step simulations, extract and save HEC-RAS results using the HEC-RAS controller, and apply MATLAB code to hydrographs or hydraulic devices.

To illustrate the performance of the interface, two case studies and three scenarios have been considered. The comparison of the accuracy yielded by manual and automated simulation approaches allows to validate the design and performance of the interface. The considered examples are representative in that several different possibilities of the MATLAB HEC-RAS interface are tested. Moreover, the proposed scripts can be easily adapted to deal with any HEC-RAS project and MATLAB control algorithm. Furthermore, documentation, tutorials, and a website have been created and made accessible to all users.

One of the most sensitive issues linked with environmental systems is that of parameter uncertainty. A typical example of this is the roughness coefficient of the river bed. Indeed, the exact value might be hard to define in certain scenarios, and tables of values are used instead, resulting in small, unavoidable errors. However, such mismatch should not result in a significant performance decrease. Moreover, the developed interface is intended to bridge the gap between the hydraulic and environmental engineering and the automatic control and AI communities, ideally leading to closer cooperation that could mitigate this issue.

On the other hand, unmodeled phenomena represent another sensitive matter to consider during the application of the integrated models with real data. Therefore, and aside from improving the quality of the model used and performing statistical filtering, the integration of other information sources (such as weather forecasts) can definitely contribute to reducing uncertainty. However, an excess of detail might also pose a challenge. Indeed, it needs to be ensured that the computational time does not exceed the sampling time, thus allowing for real-time applicability. Then, the trade-off between the level of modeling detail put in the, e.g., bathymetry and terrain, and the required computational time needs to be considered.

Short-term perspectives regard the implementation and testing of model predictive control (MPC) strategies such as those proposed in [28,29], but using an accurate HEC-RAS model of a real canal in the north of France. Such models are based on precise geographic information systems (GIS), whose data are provided by the Institut Géographique National (IGN), a French organization that manages a highly accurate geographical database of the French territory. In doing so, it will be possible to assess the performance of the MPC with a more realistic model.

Author Contributions: Conceptualization: P.S. and E.D.; methodology: R.D., P.S., and E.D.; software: R.D.; validation: R.D., P.S., and E.D.; formal analysis: R.D., P.S., and E.D.; investigation: R.D.; resources: P.S. and E.D.; data curation: R.D.; writing—original draft preparation: R.D.;

writing—review and editing: P.S. and E.D.; visualization: R.D. and P.S.; supervision: P.S. and E.D.; project administration: E.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Codes, supporting documentation and tutorials can be found at: https://sites.google.com/site/ericduviella/interface-matlab-hec-ras-documentation.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Negenborn, R.R.; van Overloop, P.J.; Keviczky, T.; Schutter, B.D. Distributed model predictive control of irrigation canals. *Netw. Heterog. Media* **2009**, *4*, 359.
2. Duviella, E.; Chiron, P.; Charbonnaud, P. A reactive control strategy for networked hydrographical system management. *Control Eng. Pract.* **2011**, *19*, 851–861.
3. Castelletti, A.; Galelli, S.; Restelli, M.; Soncini-Sessa, R. Data-driven dynamic emulation modelling for the optimal management of environmental systems. *Environ. Model. Softw.* **2012**, *34*, 30–43.
4. Ocampo-Martinez, C.; Puig, V.; Cembrano, G.; Quevedo, J. Application of predictive control strategies to the management of complex networks in the urban water cycle. *IEEE Control Syst. Mag.* **2013**, *33*, 15–41.
5. Fele, F.; Maestre, J.M.; Hashemy, S.M.; Muñoz de la Peña, D.; Camacho, E.F. Coalitional model predictive control of an irrigation canal. *J. Process Control* **2014**, *24*, 314–325.
6. Bolea, Y.; Puig, V.; Blesa, J. Linear parameter varying modeling and identification for real-time control of open-flow irrigation canals. *Environ. Model. Softw.* **2014**, *53*, 87–97.
7. Horváth, K.; Petreczky, M.; Rajaoarisoa, L.; Duviella, E.; Chuquet, K. MPC control of water level in a navigation canal—The Cuinchy-Fontinettes case study. In Proceedings of the 2014 European Control Conference (ECC), Strasbourg, France, 24–27 June 2014; pp. 1337–1342.
8. Segovia, P.; Rajaoarisoa, L.; Nejjari, F.; Duviella, E.; Puig, V. Input-Delay Model Predictive Control of Inland Waterways Considering the Backwater Effect. In Proceedings of the 2018 IEEE Conference on Control Technology and Applications (CCTA), Copenhagen, Denmark, 21–24 August 2018; pp. 589–594.
9. Baayen, J.; Becker, B.; van Heeringen, K.J.; Miltenburg, I.; Piovesan, T.; Rauw, J.; den Toom, M.; van der Wees, J. An overview of continuation methods for non-linear model predictive control of water systems. *IFAC-PapersOnLine* **2019**, *52*, 73–80.
10. Arauz, T.; Maestre, J.M.; Tian, X.; Guan, G. Design of PI Controllers for Irrigation Canals Based on Linear Matrix Inequalities. *Water* **2020**, *12*, 855.
11. Lin, N.M.; Tian, X.; Rutten, M.; Abraham, E.; Maestre, J.M.; van de Giesen, N. Multi-Objective Model Predictive Control for Real-Time Operation of a Multi-Reservoir System. *Water* **2020**, *12*, 1898, doi:10.3390/w12071898.
12. Conde, G.; Quijano, N.; Ocampo-Martinez, C. Modeling and control in open-channel irrigation systems: A review. *Annu. Rev. Control* **2021**, in press.
13. Van Overloop, P.J.; Maestre, J.M.; Sadowska, A.D.; Camacho, E.F.; De Schutter, B. Human-in-the-Loop Model Predictive Control of an Irrigation Canal. *IEEE Control Syst. Mag.* **2015**, *35*, 19–29.
14. Malaterre, P.O.; Rogers, D.C.; Schuurmans, J. Classification of Canal Control Algorithms. *J. Irrig. Drain. Eng.* **1998**, *124*, 3–10.
15. Zeng, N.; Cen, L.; Xie, Y.; Zhang, S. Nonlinear optimal control of cascaded irrigation canals with conservation law PDEs. *Control Eng. Pract.* **2020**, *100*, 104407.
16. Vermuyten, E.; Meert, P.; Wolfs, V.; Willems, P. Combining Model Predictive Control with a Reduced Genetic Algorithm for Real-Time Flood Control. *J. Water Resour. Plan. Manag.* **2018**, *144*, 04017083.
17. Duviella, E.; Hadid, B. Simulation tool of the Calais Canal implementing Logic Control based regulation. *IFAC-PapersOnLine* **2019**, *52*, 23–28.
18. Leon, A.S.; Goodell, C. Controlling HEC-RAS using MATLAB. *Environ. Model. Softw.* **2016**, *84*, 339–348.
19. Goodell, C.; Monk, S.; Lee, A.; Raeburn, R.; Karki, A.; Johnson, D. Probabilistic dam breach modeling using HEC-RAS and MCBreach. In Proceedings of the 38th Annual Conference of the United States Society on Dams, Monterey, CA, USA, 8–9 October 2018.
20. McCann, M.W.; Paxson, G. Uncertainty in Dam Failure Consequence Estimates. *E3S Web Conf.* **2016**, *7*, 11003.
21. HEC-RAS Version 5.0.7, Hydrologic Engineering Center, US Army Corps of Engineers. Available online: https://www.hec.usace.army.mil/software/hec-ras/ (accessed on 12/02/2021).
22. Goodell, C. *Breaking the HEC-RAS Code: A User's Guide to Automating HEC-RAS*; h2ls: Portland, Oregon, 2014.
23. Goodell, C. HEC-RAS File Types. 2013. Available online: https://www.kleinschmidtgroup.com/ras-post/hec-ras-file-types/ (accessed on 12/02/2021).

24. *MATLAB Version 8.6.0.267246 (R2015b)*; The Mathworks, Inc.: Natick, MA, USA, 2015.
25. *MATLAB Version 9.8.0.1417392 (R2020a) Update 4*; The Mathworks, Inc.: Natick, MA, USA, 2020.
26. Goodell, C. Controlling HEC-RAS Using MATLAB. This Blog Post Contains Links to MATLAB Code from Leon and Goodell, 2016. Available online: https://www.kleinschmidtgroup.com/ras-post/controlling-hec-ras-using-matlab/ (accessed on 12/02/2021).
27. Deshays, R. *MATLAB HEC-RAS Interface Documentation*, 2021. Available online: https://sites.google.com/site/ericduviella/interface-matlab-hec-ras-documentation (accessed on 12/02/2021).
28. Segovia, P.; Rajaoarisoa, L.; Nejjari, F.; Duviella, E.; Puig, V. Model predictive control and moving horizon estimation for water level regulation in inland waterways. *J. Process Control* **2019**, *76*, 1–14.
29. Segovia, P.; Puig, V.; Duviella, E.; Etienne, L. Distributed model predictive control using optimality condition decomposition and community detection. *J. Process Control* **2021**, *99*, 54–68.