Interactive Augmented Reality Sandbox for High-Performance Hydrodynamic Simulation and Real-Time Topographical Visualisation

Macauley Ward
University College
May 2025
Supervisor: Dr Anne Reinarz,
Department of Computer Science, Durham University

Submitted in partial fulfilment of the requirements for
the degree of *Master of Engineering in Computer Science*

Abstract—This project investigates the design and implementation of an interactive augmented reality (AR) sandbox that integrates an Intel RealSense depth camera, a projector, and a physical sandbox to provide a real-time visualisation and simulation of wave dynamics and topographical map changes, creating an engaging educational platform for exploring hydrodynamic phenomena in public outreach and research settings. High-fidelity wave and fluid simulations delivered by the parallel-capable ExaHyPE engine, enable modelling of complex hydrodynamic scenarios such as tsunamis within the interactive environment . A novel computer vision-based calibration technique utilises the Intel camera to automatically align projected visuals with the physical sandbox, enhancing setup efficiency and supporting pre-fabricated components increasing accessibility. The project highlights the role of computer vision in optimising AR sandbox systems and identifies key areas for future improvement in simulation fidelity, hardware integration, and user interaction.

Index Terms—Virtual and augmented reality, sandbox, hydrodynamics, topography, computer vision, high-performance computing, visualisation, interactive systems, educational simulations.

1 Introduction

Augmented Reality (AR) sandboxes exemplify a compelling interdisciplinary convergence of computer science, geoscience, human-computer interaction, and educational technology. Popularised by their vivid visualisation of topographical and fluid phenomena, these systems have gained traction in both academic and public outreach settings, with commerical options and military applications also explored [1]. Their success lies in their ability to translate complex scientific concepts such as hydrodynamics, topography, erosion and many other geoscientific concepts into engaging, hands-on experiences. By integrating depth sensing, real-time graphics rendering, and physical interaction, AR sandboxes draw extensively on methods from computer science and computer vision to deliver an embodied augmented learning environment. This approach facilitates intuitive user engagement and helps bridge the gap between abstract geoscientific theory and tangible, experiential understanding.

This project revisits the AR sandbox concept from a computer science perspective with the aim of enhancing the overall workflow of the system through the introduction of automated calibration, and novel depth-sensing hardware. The project will also explore the integration of research-driven high-performance computing techniques, specifically the ExaHyPE engine [2], to deliver a more robust and accurate simulation able to better simulate more complex hydrodynamic phenomena.

1.1 Background

The precise origin of the AR sandbox concept remains unclear, with Kreylos et al. acknowledging the existence of an earlier, undocumented example [3]. However, their highly influential open-source implementation at UC Davis [4] has since become the definitive reference point, catalysing significant research and development. This system has been widely adopted in geoscientific education and public outreach due to its intuitive interaction model and educational potential spawning an open-source community initiative with many AR sandboxes being developed at other institutions.

Subsequent research has explored both the pedagogical effectiveness of AR sandboxes [5], [6] [7] [8] [9] [10] [11] in educational settings and the technical expansion of the platform's capabilities, such as more accurate terrain modelling and improved environmental emulation [12] incorporating techniques such as erosion and alternative fluid sources. There are limited examples of the fundamental software and hardware components being re-evaluated, with the majority of projects building upon the existing UC Davis codebase and hardware specifications [4]. Zhou [13] has explored the use of a modern depth camera (Apple's LiDAR) and a GPU-accelerated wave solver [14] to enable the AR sandbox to be delivered in a mobile application; representing exploration of an alternative to the established base. This project will continue in this spirit and seek to achieve the following objectives.

1.2 Project Objectives & Scope

Departing from the conventional approach of re-implementing existing AR sandbox frameworks, this project will re-examine the AR sandbox from a computer science perspective, with the aim of developing a new work-flow integrating automated calibration, high-performance computing techniques delivered by the ExaHyPE engine [2] and a new depth camera (Intel RealSense D435i), whilst also exploring new physical setups. However this project will not depart from the strong conceptual foundation of the UC Davis AR sandbox [4] and the existing

open-source community [12] implementations of the same. Likewise this project will not seek to evaluate the pedagogical value of the AR sandbox, as this has been extensively covered in the literature [5] [6] and is outside the scope of this computer science project.

1.2.1 Project Deliverables

With this aim, the project will seek to deliver two key components:

- **Virtual Software Sandbox** - A fully-featured software system providing topographical visualisation and projection capabilities, forming the core functionality of the AR sandbox. It will include an ExaHyPE simulation client, depth image processing algorithms, and automated calibration routines, along with support for user interaction.
- **Physical AR Sandbox** - A tangible DIY pre-fabricated AR sandbox capable of delivering the output of the software component. Enabling vivid and engaging visualisation for hydrodynamics alongside informational visualisation of the topography and simulation.

2 Related Work

2.1 Introduction

Integration of augmented reality (AR) into physical space has opened new avenues for interactive and immersive learning experiences these are evident in the active development of the AR sandbox used to visualise and teach topography and water flow in outreach and academic settings [15] [5] [12]. This research prospect has formed the basis of many papers exploring its application and pedagogical value [5], along with the development of new features [12] to better emulate real environments and hydrological events in an AR setting. Since this paper will seek to iterate many of the underlying components of the AR sandbox, it is important to understand the existing literature whilst also identifying areas where computer vision and high-performance computing techniques can be applied to enhance the system; improving the overall workflow to constitute a novel interpretation of the AR sandbox. Accordingly, the literature will be reviewed across the following key areas: existing augmented reality sandbox systems; depth sensing and image processing techniques; calibration, alignment, and projection methodologies; hydrodynamic modelling in current applications; and high-performance wave simulation methods within research contexts to derive where improvement can be made.

2.2 Augmented Reality Sandboxes

2.2.1 Existing Augmented Reality Sandboxes

Most AR sandbox efforts have coalesced primarily around a single project developed by *University of California, Davis* (UC Davis) [4] formalised in [16] [17] although this does not appear to be the first manifestation of this idea, with Kreylos et al. acknowledging an earlier undocumented *YouTube* video [3].

The project enjoys a small active open-source community, with the original UC Davis codebase being actively maintained by Kreylos [4] [18], with many institutions building their own [19]. A focus on affordable and accessible hardware has also contributed towards its development [4]. Its success has also led to the development of commercial alternatives (e.g. [20]) to the DIY approach and thus more suited to museums and fixed exhibits, supplying a manufactured product utilising the UC Davis codebase [18].

2.2.2 Physical Construction & Specifications

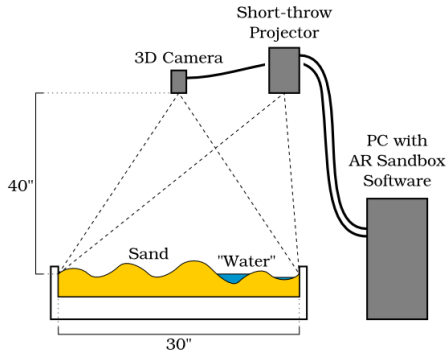


Fig. 1: Simple Diagram of the UC Davis AR Sandbox from project website [4].

Principally these projects follow the same core construction combining a physical sandbox, with a depth camera (e.g. Microsoft Kinect) and projector mounted directly above [4] [18] (see Fig. 1). The depth camera then captures the topography formed by the surface of the sand, which is then processed by software to simulate water flow with varying methods, this visualisation usually enhanced with engaging visuals is then projected back onto the sand surface in real-time [4]. This approach creates a vivid interactive simulation that can be used to demonstrate a variety of concepts [15].

2.2.3 Depth Sensing and Image Processing Techniques

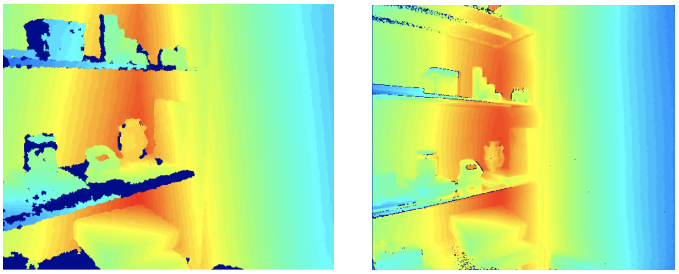


Fig. 2: Depth image from Microsoft Kinect camera v1 (640×480) & v2 (512×424) exhibited by Wasenmüller et al. in their comparative analysis [21].

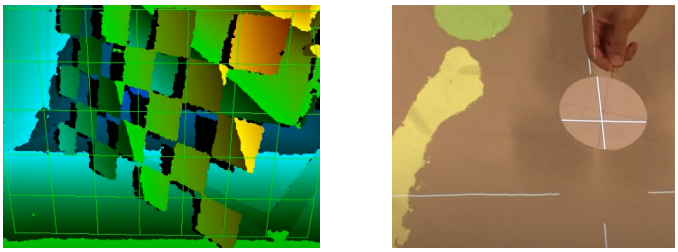
The depth camera is the core component of the AR sandbox, the UC Davis project makes use of the now discontinued although robust and capable Microsoft Kinect specifically version 1 with version 2 only experimentally supported [4]. Wasenmüller et al. explore the difference between these two cameras comparing their respective qualities [21]. As observed in Figure 2, the Kinect v1 offers a slightly higher resolution depth image (640×480) compared to the v2 model (512×424) but the quality of the v2 model is vastly superior, but both only support a standard definition output; these ultimately translate into the projection quality

notwithstanding any image processing that may take place [21]. However the reasonable second-hand cost of the Kinect makes it ideal for a budget AR sandbox setup.

Progress on alternatives to the Kinect within the UC Davis project appears limited despite the potential gains in quality that could feed into simulations of increased fidelity. Although novel work by Zhou utilises *Apple*'s integrated LiDAR camera [13] enabling deployment through an app on an *iPad* to achieve comparable functionality in a mobile package.

Kreylos et al. propose the potential utilisation of the Intel RealSense camera [4], a modern depth-capable camera with expanded capabilities and higher resolution depth and colour images (1280×720 & 1920×1080) compared to the Kinect's 640×480 [4] [22] [23]. Having been developed for and deployed in computer vision applications with an open-source software development kit, the RealSense camera provides an ethos-aligned coherent alternative to the Kinect with the Kinect having been developed for proprietary Microsoft Xbox use. However a significantly higher price point poses a challenge to the adoption of the RealSense camera in DIY AR sandbox projects, although substantial reductions in other component prices may enable its adoption. Some partial implementations with RealSense camera exist as software repositories without accompanying literature [24], but are devoid of the simulation and other key properties that constitute the AR sandbox.

2.2.4 Calibration, Alignment & Projection Methods



(a) Semi-transparent grid for intrinsic calibration

(b) Projected disc for extrinsic calibration

Fig. 3: Two examples of calibration techniques used in the UC Davis project [25] [4].

Kreylos et al. obtain a mesh from the depth camera by resolving camera and world positions within their software, this operation requires precise calibration of the camera deriving the extrinsic and intrinsic matrices required for the transformation [4] [25]. Intrinsic calibration; the process of determining the camera's internal parameters, is achieved through the use of a semi-transparent grid of known dimensions (Fig. 3a). Extrinsic calibration is achieved through the projection of a grid pattern upon the sand surface and a target disc (Fig. 3b) [4] [25]. The calibration process is essential to ensure the projected image aligns with the sand surface, and thus the simulation is accurate. Further the UC Davies project outlines some core principals in this regard, such as ensuring orthogonality of the projection image and the sand surface, as projectors may not necessarily project at a perfect $90°$ angle from the lens (throw-angle) [4]. Likewise there are some deficiencies with the factory calibration of the Microsoft Kinect Camera that result in a less favourable depth image. However the hardware is capable when accompanied by the aforementioned intrinsic calibration procedures [4].

2.3 Hydrodynamic Modelling

2.3.1 Shallow Water Equations

Hydrodynamic modelling is achieved throughout the surveyed literature by the use of the Shallow Water Equations (SWE) [4] [12] [13] which are a set of partial differential equations (PDEs) used to model fluid dynamics in shallow flow regimes; being derived from the Navier-Stokes equations which generally govern the flow of incompressible fluids, grounded in fundamental principles of mass and momentum conservation:

$$\frac{\delta \boldsymbol{u}}{\delta t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\frac{\nabla P}{\rho} + \nu \nabla^2 \boldsymbol{u},$$

where $\boldsymbol{u}$ fluid velocity vector P is the fluid pressure, ρ is the fluid density and ν is the kinematic viscosity and ∇^2 is the Laplacian operator [26] with an incompressibility condition $\nabla \boldsymbol{u} = 0$.

The Saint-Venant equations are a depth-integrated one-dimensional form of the Navier-Stokes equations, used to model the flow of shallow water where vertical depth is negligible when compared to the horizontal scale, favouring horizontal velocities. Vertical pressures are simplified with the assumption of hydrostatic pressure gradients [27], depth-averaging the horizontal plane. This depth-averaging results in a two-dimension model, with the flow expressed in the x, y plane. These characteristics are found in lakes, rivers and coastal areas, where the flow is shallow and the horizontal scale is much larger than the vertical scale, and thus suitable for the AR sandbox [27].

The first part of the Saint-Venant equations is the continuity equation, which represents the conservation of mass in a fluid system. It defines the relationship between the rate of change in water depth and the net flux (inflow and outflow) of water, and is given by:

$$\frac{\delta h}{\delta t} + \frac{\delta hu}{\delta x} + \frac{\delta hv}{\delta y} = 0,$$

where $h(x, y, t)$ is the water depth, $u(x, y, t)$ and v(x, y, t) are the depth-averaged horizontal velocities in the x and y directions respectively [27].

The second part of the Saint-Venant equations is the momentum equation, which represents the conservation of momentum within the fluid system. It accounts for the effects of hydrostatic pressures ($\frac{1}{2}gh^2$) and advection (hu^2), describing how momentum is transferred and changes over time and space, and is expressed as:

$$\frac{\partial (hu)}{\partial t} + \frac{\partial \left(hu^2 + \frac{1}{2}gh^2\right)}{\partial x} + \frac{\partial (huv)}{\partial y} = -gh\frac{\partial z_b}{\partial x},$$

$$\frac{\partial (hv)}{\partial t} + \frac{\partial (huv)}{\partial x} + \frac{\partial \left(hv^2 + \frac{1}{2}gh^2\right)}{\partial y} = -gh\frac{\partial z_b}{\partial y},$$

where g is the acceleration due to gravity, z_b is the bed elevation [27] [28] [13]. The right hand side expresses the effect of the bed elevation on the momentum ($-gh\frac{\partial z_b}{\partial y}$) but further terms may be introduced:

2.3.2 *Expanded Models (Additional Terms)*

Such terms account for the additional forces acting upon the fluid, enhancing the realism of the simulation by better emulating real-world hydrological phenomena. These could include friction, turbulence, wind stress and other forces such as the Coriolis force [12] [13].

Smith et al. expanded the functionality of the sandbox by implementing point sources, with the aim to facilitate the exploration of water routing, meandering rivers, erosion, river rise due to run-off [12]. Additionally *Continual waves* were implemented; modelling ocean waves through basic sinusoids [12]. Ground absorption is also modelled [12].

Alternatively Boussinesq equations propose an important extension of the SWE, accounting for dispersion and non-hydrostatic pressure effects, prominent in modelling wave propagation in coastal areas, shoals, shallow seas and harbours [29] [30] [13]. Differing from the SWE, which assume hydrostatic pressure and neglect vertical acceleration, the Boussinesq equations allow for a more accurate simulation of these more complex wave behaviours.

Zhou implements Boussinesq equations using a GPU-accelerated solver developed by Tavakkol et al. [14], specifically optimised for real-time interactive simulation and visualisation using GPU acceleration in an open source and mobile package [13]. Enhancing the accuracy of the AR sandbox in the aforementioned areas where SWE provide only a simplified model. Zhou extends this work into further augmented reality application beyond the sandbox, with a focus on coastal applications [13].

2.3.3 *Model Computation*

Whilst the adoption of SWE is unanimous, the manner of computation is not with a number of methods being trailed; however the basis for these methods is consistent consisting of: a method of discretisation, a time-stepping scheme, implementation of boundary conditions and a stability condition.

The UC Davis project in particular implements a standard finite volume method to discretise the PDEs, with a second-order Runge-Kutta time-stepping scheme [4] (although Euler is also explored), and a Courant-Friedrichs-Lewy (CFL) stability condition [4].

Zhou by implementing the work of Tavakkol et al. inherits a more complex time-stepping scheme consisting of multiple steps such as third order Adams-Bashforth & Adams-Moutlon schemes [13] [14]. The discretisation strategy (FV) however remains the same, highlighting the alternative configurations that can be explored.

Both the UC Davis project and Zhou's work utilise GPU acceleration on both the wave solver and rendering portion of the software using OpenGL (GLSL) and Apple's Metal Shading Language (MSL) respectively [13] [4]. This is essential for real-time performance whilst maximising the computational performance; although CPU based HPC solutions do exist often drawing on parallelism.

2.3.4 *Wetting & Drying*

An integral part of the simulation is a realistic interface between the "wet" and "dry" regions of the simulation delineated by the fluid body and the terrain. However due to the number of complex hydrological phenomena that can occur in shallow water especially in the run-up to land, these can be difficult to model. A naive implementation presents a risk of numerical instability and degrading of the simulation, whilst a complex scheme could introduce unnecessary computational burden. For this reason a simple thresholding approach is often adopted, where a threshold is set to determine the wetting and drying of the terrain [4] [12]:

$$\text{If } h < \epsilon \text{ (dry), then } h = 0,\ u = 0,\ v = 0,$$

where h is the water depth, u and v are the horizontal velocities in the x and y directions respectively, and ϵ is a small threshold value [4] [12].

This approach is simple to implement and computationally efficient but is subject to the deficiencies described.

Alternatively a well-balanced scheme as in Zhou may be adopted where a pressure balance is sought between the water body and the topography through hydrostatic reconstruction especially when at rest (ie a lake) [13] [31].

2.4 Software & Visualisation

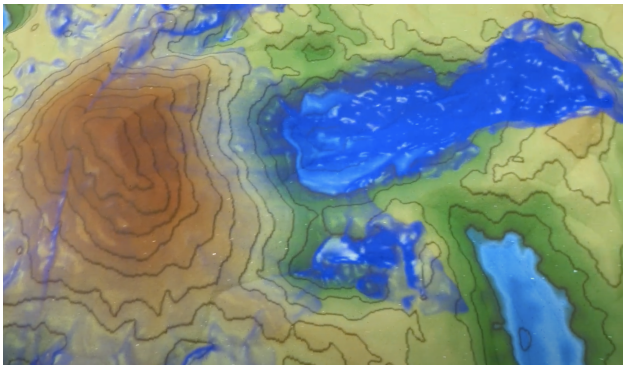


Fig. 4: UC Davis AR sandbox by Kreylos et al. [32].

Software development for the AR sandbox has been primarily driven by the UC Davis project [4], with Kreylos et al. providing an open-source project based off their own original computer vision framework written in C++. This framework interfaces with `OpenGL` to deliver the visualisation and simulation of their AR sandbox [4] [32] through the use of compute shaders. An example of their work is exhibited in Figure 4 exhibiting fluid flow down a mountain.

The software is run locally with an interactive GUI allowing the user to manipulate the simulation and visualisation parameters, with a number of pre-defined scenarios available to explore [4]. Hand gestures allow some degree of interaction with the simulation, although this is limited to a few gestures [4]. This does present an opportunity for methods of interaction specifically when considering its use in educational group settings.

2.5 Pedagogical Research & Learning Applications

Comprehensive work to establish the pedagogical value of the AR sandbox has been conducted by Soltis et al. [6], and to a lesser extent by Woods et al. [5].

Woods et al. performed a pilot study to evaluate the effectiveness of the AR sandbox in teaching topography

and water flow to students. The study found that the AR sandbox was effective in teaching these concepts, with students reporting a positive experience and increased understanding of the concepts with high praise indicative of an engaging tool [5]. Woods et al. attribute this effectiveness to the tangible link between 3D topography and their 2D representations within the sand box, enabling spatial learning and providing an opportunities for embodied learning exploring the behavioural science behind this [5]. Although Woods et al. acknowledge the limitations of their study; with only a small sample size taking part in the study [5].

Soltis et al. conducted a larger study with 91 participants, deploying a more rigid psychological & behavioural science perspective [6]. This methodology involved the use of four methods to evaluate the goal of the study including: skin conductance, dynamic video, a post-session test (Purdue Spatial Visualisation Test & Topographical Map Assessments) and interviews [6]. Both structured, semi-structured and unsupervised sessions were conducted to also asses the effectiveness and usability of the tool in these settings [6]. A demographics study was also conducted to uncover any determinant factors in effectiveness.

Overall the study demonstrated that students of all spatial abilities showed an increased level of engagement with the AR sandbox, with this enthusiasm being reported by the students [6]. The study also found that a structured lab approach was more effective in enhancing spatial abilities and engagement [6].

2.6 Limitations & Advancement

Much of the survey literature rigid adopts the well established UC Davis model [4], with little deviation. However there are a few areas where the application of computer vision techniques and revised workflows could potentially enhance and revise the concept.

Of particular interest is the fluid simulation itself, with Smith et al. acknowledging the existence of high fidelity computational research models [12]; that might be adapted for use within the AR Sandbox settings, with some works exploring this [33] and [13] which deploys a model described in [14]. The integration of these models into the AR sandbox provides opportunity for further research and educational interdisciplinary dialogue.

Integration of the ExaHyPE [2] engine poses a further opportunity to explore the use of high-performance computing (HPC) in the AR sandbox, an area not yet researched. This could enable the simulation of more complex hydrodynamic phenomena, such as tsunami propagation and coastal dynamics, which stress the limitation of existing approaches. Integration of an external toolkit would also allow for rapid prototyping and testing of new models, as well as the ability to leverage existing research and models within the AR sandbox without the need for intimate understanding of the existing sandbox codebase (being reliant primarily on GPU shading languages and OpenGL).

Exploration of parallelism within the simulation pipeline could also be explored, with the UC Davis project acknowledging the limitations of their hardware (NVidia GeForce 780), with high turbulent flows causing a high number of integration steps increasing the computational burden [4]. Newer models may offer some efficiencies decreasing the computational burden, allowing a higher fidelity to be achieved with comparable hardware.

A higher fidelity simulation is best served by a higher resolution depth input, not achievable with the default Kinect camera. With the discontinuation of the Kinect, a search for alternatives is justified. The Intel RealSense camera [34] or the iPad LiDAR camera [13] are viable options to explore. With a higher quality depth image and simulation enhancing the visualisation.

Moving to a mobile deployment of these techniques as explored by Zhou [13] might also offer some opportunity due to the advent of powerful mobile GPUs and depth cameras, although the trade-off in performance may be a limiting factor especially where model fidelity is considered.

Although a mobile approach may be adopted in the control software; potentially serving to address the reduced effectiveness of the sandbox in group settings as highlighted by Soltis & Woods et al. [6] [5]. Whilst a distributed approach separating the control, simulation and rendering components may offer potential benefits in terms of performance and flexibility, it may also introduce additional complexity and potential latency issues.

Calibration and alignment are another area where improvements could be made, allowing the introduction of more ad-hoc pre-fabricated cost effective components. In contrast to the UC Davis project which relies on accurate calibration and physical alignment of the projector and depth camera [4]. This is a significant barrier to entry for many users with several video tutorials attempting to address this. A more dynamic approach may increase accessibility and ease of use.

3 Proposed Methodology

3.1 Physical Design & Construction

3.1.1 System Overview

Figure 1 & 5 illustrate the key components of the AR sandbox system, including the projector, depth camera, and sandbox. The projector is mounted above the sandbox, projecting images onto the sand surface. Along with the depth camera positioned to capture the sand surface and its topography. The sandbox is filled with sand, which can be manipulated by users to create different topographies which are then visualised. A number of considerations are to be made when selecting the required components to ensure that they are suitable for the intended purpose and can be easily integrated into the system.

3.1.2 Hardware Selection & Design Requirements

Kreylos et al. propose a DIY approach to the AR sandbox, with the aim of making it accessible to a wider audience [4]. However, their sandbox consists primarily of constructed materials such as wood, which incur an immediate cost and may present barriers to those without access to tools or construction skills. Research configurations such as those found in Zhou, are more accessible with the use of pre-fabricated components for the frame. Therefore the proposed design will be constructed of primarily sourced pre-fabricated components such as a plastic tub and existing camera tripods & mountings. These components should simplify the design, lowering costs and complexity, while enhancing accessibility through a modular structure that facilitates assembly, storage, and transport. The following components are proposed for the design:

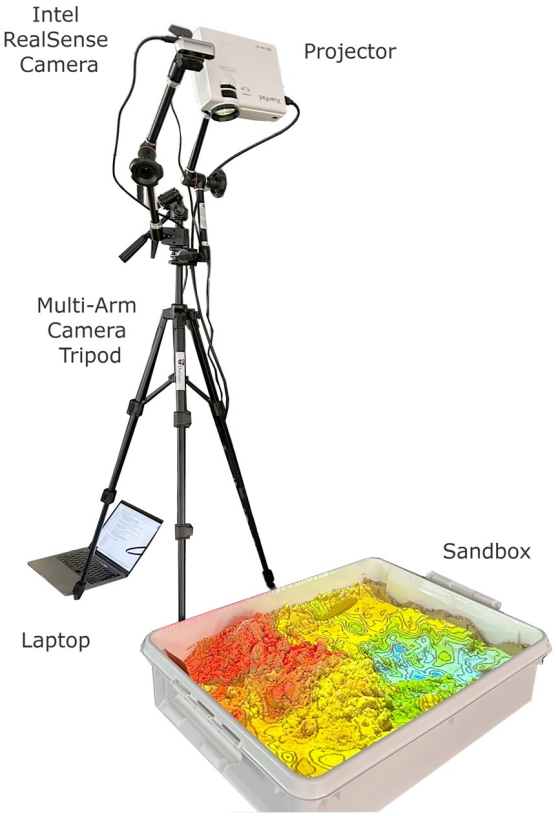


Fig. 5: Physical setup of the sandbox system.

- **Sandbox** - A plastic tub with a removable lid, which can be filled with sand. The lid will be used to cover the sandbox when not in use, preventing spillage and contamination. The tub should be large enough to accommodate a sufficient amount of sand for the visualisation whilst being reasonably portable. The UC Davis projects uses a container of size $1 \times 0.75 \times 0.1$ m [4], however in this case a smaller container will be sought for convenience and portability. A height-to-width ratio of $4:3$ shall be maintained to best match standard projector aspect ratios. The sandbox shall be filled with standard play sand; which is easily available and inexpensive; in contrast to the white sand often used [4] [20].
- **Projector** - A short-throw projector capable of projecting high-resolution images onto the sand surface whilst not incurring excessive distortion. Projectors of this type are available for under £100, with a minimum throw distance of 1m. It must be mounted securely above the sandbox and aligned precisely to project visuals accurately onto the sand surface, minimising distortion and maximising clarity.
- **Mounting Hardware** - Tripods and brackets to securely mount the projector and depth camera above the sandbox. The mounting hardware should be adjustable to allow for precise alignment of the projector and camera. Both the camera and projector should be mounted close to each other, preferably orthogonal to the sand surface, to ensure accurate calibration and projection. Additionally, weight distribution and safety should be carefully considered to prevent tipping or instability during use.
- **Computer** - Moving away from the UC Davis project that requires a mid-range computer (Intel i5 3GHz, GeForce GTX 1060) [4], a laptop with an integrated GPU of moderate specification will be utilised moving towards the completely mobile approach proposed by Zhou [13]. This should be capable of running the simulation and visualisation software, as well as processing the depth images from the camera with sufficient performance.
- **Depth Camera** - An Intel RealSense product will be selected for the project, replacing the commonly used Kinect camera [4] [12]. The RealSense camera is a more modern and capable depth camera, offering higher resolution depth images and improved accuracy. It is also more compact and easier to integrate into the system. A USB 3.0 connection will be used for maximum bandwidth and performance.

3.1.3 Physical Design Evaluation

Each component of the design will be evaluated based on its effectiveness in achieving the goals of the project, and on meeting the prescribed criteria of being: pre-fabricated, easily setup, low-cost and fully functional.

3.2 Intel RealSense D435i Camera Integration

The Intel RealSense D435i camera was selected for this project due to its advanced depth-sensing capabilities, compact design and primarily its compatibility with modern computer vision workflows and libraries. Dual capture of RGB and depth images at a greater resolution than the Kinect camera, offers improved accuracy and detail that should translate through to a high fidelity visualisation. The D435i camera is also equipped with its own robust calibration and alignment features that will aid in the automated calibration process to be proposed. Additionally it's comprehensive SDK will facilitate easy integration with the software and hardware components of the project, allowing for rapid prototyping and development; whilst also removing the compatibility issues and deprecated state of the proprietary Kinect camera. This upgrade in technology should enhance the overall performance and reliability of the AR sandbox system. The accompanying library `librealsense` also provides a range of tools that can be used to record, process and visualise the images for development and later testing purposes.

3.3 Automatic Calibration and Alignment

Precise alignment between the projected visualisation and the physical sandbox is crucial for accurate visualisation upon the sand surface. Without this alignment, the projected image may not accurately represent the physical terrain, leading to a disjointed and ineffective experience for users. The UC Davis project employs both a physical semi-transparent grid for intrinsic calibration and a target disc for extrinsic calibration [4] [25]. However, these methods can be time-consuming and may not yield optimal results in all scenarios; relying on the users ability to correctly function the software. This project proposes the application of standard computer vision techniques to automate the calibration process, improving the accuracy and efficiency of the alignment; removing the need for user.

3.3.1 Region of Interest (ROI) Extraction

Unlike the projector that can be physically aligned with the sand surface, alignment of the depth camera is often not possible due to its often wide field-of-view (FOV) and the physical constraints of the sandbox and mounting.Previous projects require the user to select a ROI from the depth image, which is then used to calibrate the projection, with some using only a fixed rectangular selection tool. This can lead to an imprecise calibration process, as the user may not select the correct area of the depth image that corresponds to the physical sand surface.

This process may be automated by using computer vision techniques to extract the ROI from the depth image, which can then utilised for calibration. This enables a more accurate calibration process, as the process is removed from the user and extracts precise corners of the sand surface.

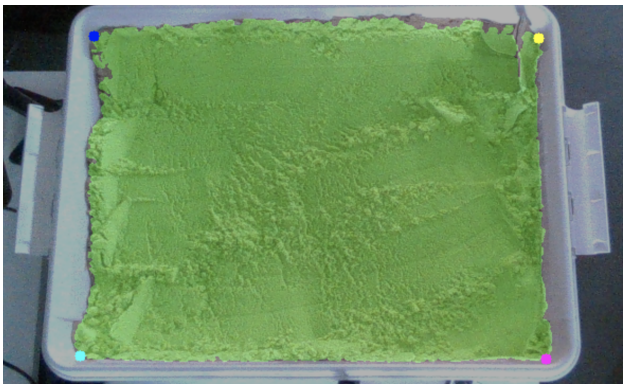


Fig. 6: Region of Interest (ROI) extraction process for automatic calibration.

3.3.1.1 Histogram Backprojection & Colour Thresholding: Selection of the ROI (the sand surface) shall be achieved by considering the colour distribution within the colour image captured by the depth camera. The colour image shall then be converted to the HSV colour space; removing the luminance component and allowing for more robust colour-based segmentation. A region that is either selected or considered from the centre of the image will then be used to create a histogram of the target distribution (the distribution of the sand). By discarding the luminance component, the histogram will be more robust to changes in lighting conditions and shadows, which can affect the colour distribution of the sand.

The histogram will then be back-projected onto the complete image, identifying pixels that match the sand colour distribution. Once identified the pixels will be thresholded to create a binary mask of the sand; the further processing of which shall derive the ROI.

3.3.1.2 Dilation, Contouring, & Masking: As the back-projection process identifies pixels that match the sand colour distribution, it is likely that the mask will contain noise and small holes. To address this, morphological operations such as dilation and closing can be applied to the binary mask to remove noise and fill in small holes.

A distinct rectangular region should then be present within the mask, which can be extracted using contour detection techniques. It is then assumed that the largest contour corresponds to the sand surface previously identified; with smaller regions being discarded. A new mask of the largest contour is then present, which can be used to extract the ROI from the original depth image.

3.3.1.3 Extreme Point Detection & Perspective Warping: The visualisation output will be projected upon the surface through an OpenGL shader; requiring that the texture coordinates of the image are known and suitably calibrated. This will require that corners of the ROI which define the rectangular region are known. A simple loop through contour points can be used to find the extreme points in the x and y directions, which correspond to the top-left, top-right, bottom-left, and bottom-right corners of the ROI.

3.3.2 Homography Estimation

Fig. 7: Checkerboard pattern used for homography estimation and calibration.

Once derived the extreme points must be projected from the camera space to the projector space to enable correct alignment of the projected image with the physical sandbox. This requires a perspective transformation between the two planes of the camera view and the projector display; defined by a 3×3 homography matrix $\boldsymbol{H}$ satisfying the equation:

$$\boldsymbol{x}_i' \sim \boldsymbol{H}\boldsymbol{x}_i,$$

where $\boldsymbol{x}_i' = (x_i', y_i')$ is the point in camera space, $\boldsymbol{x}_i = (x_i, y_i)$ is the corresponding point in projector space.

Estimation of the homography matrix can be achieved by considering a set of corresponding points between the two planes. A checkerboard pattern is commonly deployed in computer vision settings for this purpose and shall be deployed in this case, with a pattern projected upon the sand surface (see Figure 7). This differs from the semi-transparent grid used for Kinect calibration in the UC Davies as it is manifest through the projector rather than a separate specific tool.

Internal intersecting corners of the checkerboard pattern are used to define the corresponding points between the two planes, with the coordinates of these points in projector space already known, the camera space points can then be visually identified within the image; aided by the simple alternating pattern of the grid. With both sets of corresponding points known, a minimum of four points are taken to estimate the homography matrix $\boldsymbol{H}$ using a direct linear transformation (DLT) formulation:

$$\boldsymbol{x}_i' \times \boldsymbol{H}\boldsymbol{x}_i = 0,$$

providing two linearly independent equations per point pair:

$$\begin{bmatrix} x_i' & y_i' & 1 & 0 & 0 & 0 & -x_i x_i' & -y_i x_i' & -x_i \\ 0 & 0 & 0 & x_i' & y_i' & 1 & -x_i y_i' & -y_i y_i' & -y_i \end{bmatrix}$$

these are built into a matrix $A \in \mathbb{R}^{2n \times 9}$, where n is the number of point pairs. The homography matrix $\boldsymbol{H}$ can then be estimated using Singular Value Decomposition (SVD) on the matrix solving $\boldsymbol{A}\boldsymbol{h} = 0$. The matrix is obtained by reshaping $\boldsymbol{h}$ into $\boldsymbol{H}$. These functions shall be implemented using `OpenCV` [35].

Once the extreme points are projected they are normalised within the screen space to provide the texture coordinates for the OpenGL shader.

3.4 ExaHyPE Wave Solver Integration

3.4.1 ExaHyPE for Hydrodynamic Modelling

The Exascale Hyperbolic PDE Engine (ExaHyPE) by Reinarz et al. describes its purpose as a software engine for solving systems of first-order hyperbolic PDEs such as the shallow water equations (SWE) [2] (& [36]). It's primary distinction is the ability to provide a high-performance scalable framework allowing models to be implemented and executed across a range of hardware platforms; from a "student's laptop to a supercomputer" [2], whilst removing some of the particulars regarding parallelism and high-performance computing (HPC) rather shifting the focus to the models and scenarios. The engine is generalised allow users to tailor it to their specific needs, focusing on the numerical methods and solvers used to solve the PDEs, which are accommodated through a modular architecture. Other capabilities include adaptive mesh refinement, which allows for the simulation of complex geometries and phenomena.

Reinarz et al. demonstrate the capabilities of ExaHyPE through a number of examples across the fields of hydrodynamics, seismology and astrophysics [2]. Their SWE example and further work on Tsunami modelling [36] is of particular interest, as it demonstrates the capabilities of ExaHyPE in simulating complex hydrodynamic phenomena. It is anticipated that these capabilities can be downscaled and integrated into the AR sandbox to enable more accurate and efficient simulations of wave propagation and other hydrodynamic phenomena by way of its modular architecture. Moreover, by moving beyond the fixed shader defined models as commonly explored in literature, this integration will provide a more flexible and extensible framework for simulating complex scenarios. This advancement not only enhances the educational value of the system but also creates opportunities for continued iteration and development.

3.4.2 Solver Configuration

Configuring ExaHyPE for hydrodynamic modelling follows the example provided by Reinarz et al. for their simulation of the 2011 Tohoku Tsunami [2]. Adopting a modular engine concept - the numerical methods are pre-defined with the user providing a kernel that implements the specific PDEs to be solved. The kernel is then compiled alongside the engine using a layer of auto-generated code abstracting the kernel implementations, allowing users to focus on model configuration.

The engine supports two principal numerical schemes: a finite volume method and the ADER-DG (Arbitrary high-order DERivative Discontinuous Galerkin) method. The finite volume method works with average values in discretised cells and calculate fluxes across cell boundaries. The ADER-DB method represents the solution as high-order polynomials within each cell using a two-step process: first predicting the solution inside each cell, then correcting it using information in neighbouring cells.

Mesh discretisation is based on an adaptive Cartesian grid in two dimensions managed by the Peano framework. Mesh refinement uses a tri-partitioning approach, improving performance and memory management.

Time-step sizes are dynamically adjusted during simulations to satisfy the CFL condition for stability, with constraints depending on polynomial order, grid resolution, and local wave speeds.

3.4.3 Shallow Water Equations

The shallow water equations are realised within the ExaHyPe engine by first implementing the flux terms; representing the flow of conserved quantities (both mass and momentum) across the cell boundaries. In the context of SWE, these fluxes are derived from the conservation laws previously described; and are manifest in the water height and both horizontal momentum fluxes: h, hu, hv.

$$\frac{\partial}{\partial t}\begin{pmatrix} h \\ hu \\ hv \\ b \end{pmatrix} + \nabla \cdot \begin{pmatrix} hu & hv \\ hu^2 & huv \\ huv & hv^2 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 \\ gh\,\partial_x(b+h) \\ gh\,\partial_y(b+h) \\ 0 \end{pmatrix} = 0,$$

from [2, Eq. 2], where h is the water height, hu and hv are the horizontal momentum fluxes in the x and y directions respectively, and b is the bathymetry.

The flux function $\nabla \cdot F(Q)$ is is computed internally by the solver according to its specification through the numerical schemes internally implemented: the finite volume (FV) method or the ADER-DG method as previously described.

The source terms, which account for the effect of gravity and varying bottom topography, are included in the weak formulation and evaluated as part of the local cell updates. These are particularly important in scenarios with strong bathymetric gradients, such as tsunami simulations or flow over uneven terrain. In ExaHyPE, both the flux and source terms are handled through user-defined or default kernel functions, which are invoked automatically as part of the solver pipeline.

The SWE module in ExaHyPE provides default implementations for the flux function and Riemann solver, but experimentation with alternative formulations will be explored to examine certain scenarios within the AR Sandbox.

3.4.4 Data Pipeline and Integration Workflow

Integration of the ExaHyPE engine into this project requires a robust data pipeline to facilitate the exchange of information between the simulation engine and the visualisation framework. The differing components of the system must communicate effectively to ensure that the simulation results are accurately represented in the visualisation and processed in reasonable

time.This workflow incorporates several key tools & formats to achieve this:

3.4.4.1 Data Formats (NetCDF & VTK): Two structured data formats facilitate data transfer between the simulation and visualisation components: NetCDF and VTK. Both formats are widely used in scientific computing and integrate well with applications such as ParaView and are supported by the ExaHyPE engine.

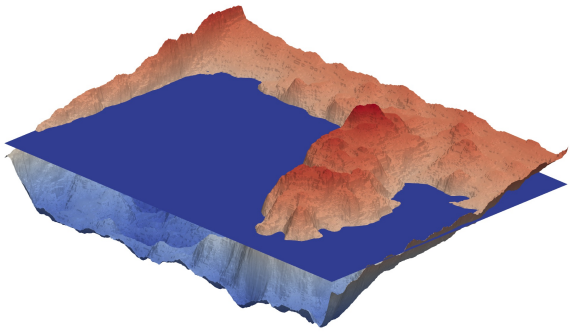


Fig. 8: Illustration of the two simulation input components as viewed in ParaView.

NetCDF is a binary format that is efficient for gridded data structures or storing large datasets, and shall be used for the simulation input. Two components comprise the input data and define the initial conditions for the simulation: the bathymetry and the initial water height (see Fig. 8), both represented as 2D grids of fixed resolution directly determined by the depth camera resolution so that no data is lost.

VTK is a more flexible plain-text format that supports a variety of data types and structures, and shall be used for the simulation output. The simulation output is multi-faceted consisting of the conservative variables (water height, momentum fluxes) and the bathymetry, but also time and mesh refinement information, necessitating a more flexible format. The output is provided as a series of time steps, which shall require reconstruction upon visualisation.

3.4.4.2 Docker Containerisation: To ensure reproducibility and simplify deployment and compilation, the ExaHyPE engine is containerised within a Docker image. This allows for easy distribution and deployment of the engine across different platforms, ensuring that all dependencies are included and that the engine can be run in a consistent environment.

A downside of the containerised approach is that the simulation may be removed from the local machine and deployed on a remote server. A small lightweight server is therefore required to be deployed to facilitate communication between the simulation and visualisation components.

3.4.5 Challenges and Mitigation Strategies

Despite its capabilities, integrating ExaHyPE into a broader interactive system—such as an AR sandbox introduces several challenges. These span computational performance, system integration, stability, and usability. This section outlines key issues and the strategies used to address them.

3.4.5.1 Computational Performance: Computation of complex hydrodynamic phenomena can be computationally intensive depending on a number of factors including the size of the simulation domain, the resolution of the mesh, and the complexity of the model. The ExaHyPE engine is designed to be highly efficient and scalable, but performance shall remain a concern given the aim of utilising limited hardware for most-simulation, whilst enabling research exploration.

To mitigate this concern, the simulation domain will be kept small, with a suitable resolution to ensure that an insightful visualisation of the simulation output can be achieved for low hardware settings.

3.4.5.2 Latency & Real-time Simulation: ExaHyPE is designed for research purposes and thus for pre-computed simulations and not real-time applications. This runs contrary to the established real-time performance of the AR sandbox. Although due to the focus on simulating more complex phenomena rather than the standard toy simulation approach; a compromise may be made here forgoing real-time execution.

Additionally loading and processing of simulation inputs and output will incur a computation burden, which may be exacerbated by the use of a remote server. To mitigate this, the simulation will be run in the background, with buffering where applicable. Additional asynchronous triggering and processing of data where suitable will be maximised. Data shall be processed in batches of a suitable time-span.

3.4.5.3 Stability and Accuracy: With user-defined topographies derived from the depth camera, the model will be subject to a wide range of initial conditions and scenarios. This may lead to instability in the simulation, especially in cases of strong gradients or discontinuities. The ExaHyPE engine is designed to handle some of these scenarios, but care must be taken to ensure that reasonable limits on the initial conditions are set. Whilst also ensuring that these are effectively communicated to the user.

3.5 Visualisation & Software Architecture

3.5.1 Software Overview

To facilitate the projection of the simulation output alongside a topographical map, a visualisation pipeline is required. This shall be delivered through a custom-built software application that integrates the various components of the system. The software will be responsible for processing the depth images, extracting the ROI, calibrating the projection, interfacing with the ExaHyPE simulation interface, and rendering the combined output in real-time, whilst also enabling user interaction.

3.5.2 Development Environment and Technologies

The software will be developed using C++ and OpenGL, with the aim of achieving high performance and real-time rendering capabilities. Other technologies such as game engines were considered but ultimately rejected due to the need for low-level control over the hardware and the potential for the introduction of unnecessary overhead. With C++ providing a more efficient and flexible solution for the specific requirements of the project, whilst also providing access to several comprehensive libraries that shall interface all of the required components:

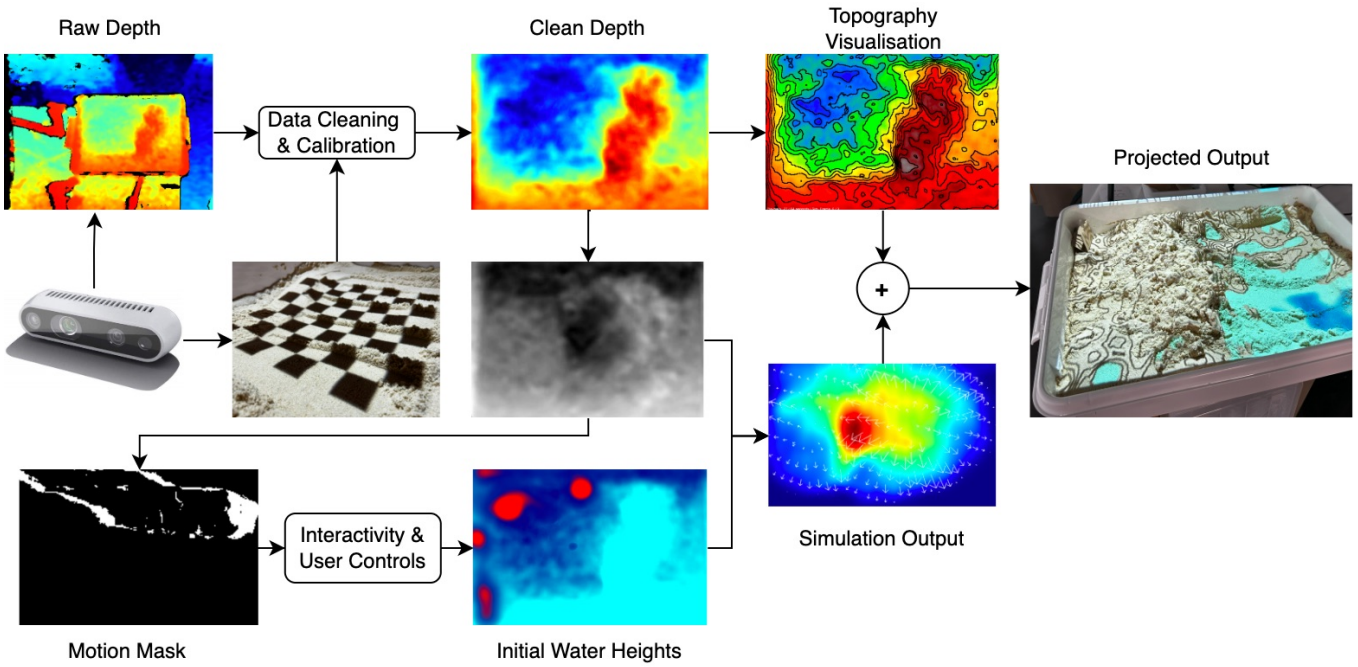


Fig. 9: Illustration of the methodology workflow.

Library	Purpose
`OpenGL & GLAD`	Core graphics API used for hardware-accelerated rendering of simulation results. GLAD provides OpenGL access and extensions.
`GLFW`	Provides cross-platform window management, OpenGL context creation, and input handling.
`GLM`	A header-only mathematics library providing vectors, matrices, and transformations.
`OpenCV`	Used for image processing tasks including depth map filtering, smoothing, and pre-simulation preprocessing.
`librealsense`	Intel SDK for accessing and managing RealSense depth camera streams and device settings.
`UM-Bridge`	Middleware interface that facilitates communication between simulation clients and external solvers like ExaHyPE.

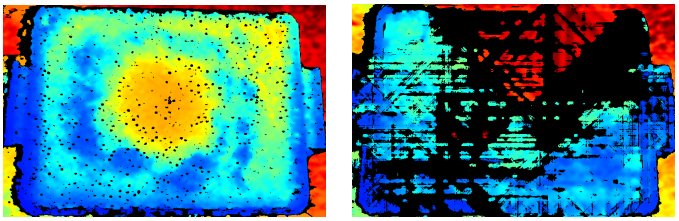
TABLE 1: Libraries used for visualisation, simulation, and data acquisition.

While C++ offers fine-grained control, it introduces complexity in memory management, pointer safety, and concurrency handling particularly in this application with several asynchronous processes, increasing the likelihood of segmentation faults and undefined behaviour unless robust practices (i.e smart points & mutexes) are followed.

To ensure cross-platform compatibility, the software will be developed using `CMake`, which simplifies the build process and allows for easy integration of external libraries.

3.5.3 Data Acquisition

First the topography of the sand surface is captured using the chosen RealSense camera. The `librealsense` library is used to access the camera streams and retrieve the depth image in addition to the RGB colour image, these are transferred through a USB 3.0 connection for sufficient bandwidth, frame rate and resolution; whilst ensuring minimal IO latency. These images are subject to the configuration of the camera [37] which shall be appropriately set to ensure a stable image in high to low light conditions.



(a) Standard Noise (b) Extreme Noise

Fig. 10: Depth images with varying levels of noise.

3.5.4 Elevation Mapping & Preprocessing

The depth image is then processed to remove temporal and spatial noise along with an missing data (holes), which are due to environmental factors such as lighting conditions, occlusions and any projector interference upon the sandbox surface. Some patterns of noise are visible in Figure 10.

3.5.4.1 Temporal Noise: Temporal noise will be addressed through the use of a low-pass pixel filter, that allows for the smoothing of the depth image over time. This is achieved by averaging the depth values of each pixel over a specified number of frames, reducing the impact of noise and fluctuations in the depth data. Two methods will be evaluated a `OpenCV` moving average filter and a persistency pixel-average filter from `librealsense`.

3.5.4.2 Holes & Spatial Noise: To address any holes in the data or spatial noise `librealsense` filters and `OpenCV` functions such as in-painting, Gaussian blurring, and morphological operations will be applied to de-noise and reasonably complete and clean the data. A variety of alternatives will be explored to determine the most effective approach for the specific application. Methods such as Gaussian blur may smooth over sharp topographical transitions; a drawback for sandbox environments where user-created ridges or valleys are small-scale but meaningful. Bilateral

filtering could retain edge information better, though at a higher computational cost. The refined 16-bit floating point depth image is cropped to exclude extraneous regions from the RealSense's wide FOV and is transferred to the GPU via OpenGL for shader-based rendering.

All image preprocessing will be done on the CPU via the described methods, with the exception of the final rendering which will be performed on the GPU via OpenGL. A fully GPU-accelerated pipeline using compute shaders could be considered, but this would require a more complex implementation and may not yield significant performance gains for the current application. Additional use of a more tangible approach in the design will aid any future development and testing of the system rather than deferring to GPU shaders that potentially obscure functionality.

3.5.5 Preprocessing & Noise Reduction Evaluation Methodology

To judge the effectiveness of the preprocessing methods in addressing temporal variance, a number of metrics will be used to evaluate the quality of the depth image. These will consider consecutive frame-to-frame differences:

- **Mean & Standard Deviation**: The mean μ and standard deviation σ of the depth image pixels are calculated providing a measure of the overall brightness and contrast stability over time.
- **Mean Squared Error (MSE)**: MSE measures the average squared difference between the consecutive frames and will be used to quantify the amount of noise in the depth image. A lower MSE indicates a more stable image.
- **Peak Signal-to-Noise Ratio (PSNR)**: PSNR will consider the ratio between the underlying signal (the depth image) and the noise (the difference between the consecutive frames); with a higher PSNR being indicative of a better temporal filtering. It is defined as:

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{M^2}{\text{MSE}}\right)$$

where $M = 65535$ the maximum value of the 16-bit depth image.

Further the Structural Similarity Index (SSIM) will be used to measure the similarity between ground-truth and processed images under a variety of conditions to asses the effectiveness of the hole filling preprocessing methods. SSIM is a perceptual metric that considers changes in structural information, luminance, and contrast. It is defined as:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

where μ_x and μ_y represent the mean pixel intensities of the images, σ_x^2 and σ_y^2 are their variances, σ_{xy} is the covariance, and C_1 and C_2 are stability constants.

Time considerations will also be made to ensure that the preprocessing methods do not introduce excessive latency into the system.

3.5.6 Simulation Integration

ExaHyPE shall be configured to accept two NetCDF input files containing the initial conditions for the simulation: the terrain / bathymetry and the initial water height. The terrain is derived from the depth image captured by the RealSense camera, with the initial water height set to another `OpenCV` image derived from user interaction (this shall be of equal resolution).

The data is passed to the ExaHyPE engine either through the `UM-Bridge` interface or through mounted directories pertaining to the Docker image. The bridge can then be used to trigger the simulation and retrieve the output data after some delay. The simulation output is provided as a series of VTK files, which are then processed and visualised in real-time.

Output processing requires loading the VTK files and extracting the relevant data for visualisation. This includes the conservative variables (water height, momentum fluxes) which shall be used to provide a range of informative visualisation. As this data is discretised over a grid, it is necessary to interpolate the data to provide a smooth visualisation. This is achieved using a Gaussian kernel (`OpenCV`) allowing smooth interpolation of the data points. This is then passed to the OpenGL shader for rendering.

The preprocessed simulation does introduce some user considerations into the design, a long turnaround time for the simulation may be required; which users may interpret as unresponsiveness or lag. A visual placeholder or loading screen may be required to indicate that the simulation is running and that the system is responsive. This could be a simple animated loading icon or a message indicating that the simulation is in progress. Additionally failure of the simulation should be properly communicated to the user, with a clear error message indicating the cause of the failure.

3.5.7 Visualisation Pipeline

The visualisation pipeline is implemented within a primary `OpenGL` shader, tasked with rendering both the simulation output and the topographical map in real-time. The displacement map and simulation output are passed to the shader as textures, which shall then be evaluated over a quad-mesh representing the sand surface, this quad shall be positioned and scaled according to the calibration process. Figure 9 illustrates the complete visualisation pipeline, from the depth image acquisition to the final rendering of the simulation output within the sandbox.

3.5.7.1 Topographical Mapping: A color gradient or map is applied to visually represent elevation, following conventions commonly used in computational topography frameworks like Generic Mapping Tools (GMT). Multiple color map options will be implemented, allowing users to dynamically select or customise the gradient to enhance clarity and engagement.

Contour lines shall also be implemented to provide a more informative visualisation of the topography, these are derived from the heightmap and are rendered as lines over the quad-mesh according to the defined height interval. To ensure crisp and defined lines, the contour lines are determined by considering a neighbourhood of four pixels and determining the maximum and minimum values within that neighbourhood; the interval is then considered as in the UC Davis project [4]. If this method proves ineffective then a more complex method such as the Marching Squares algorithm shall be considered.

3.5.7.2 Simulation Output: Once a base-color for the topographical map is established, water levels shall be visualised using dynamic overlays—ranging from semi-transparent blue surfaces to animated vector fields showing flow direction.

3.5.8 User Interaction & Motion Detection

Primarily user interaction is achieved through manipulation of the sand to form various topographies, which are then visualised in real-time. To ensure that only useful visual feedback is provided the system will not update when the sand is being manipulated, but rather when the user has finished manipulating the sand surface. This will be achieved through a simple threshold upon the temporal variance of the depth image, where if the variance of the depth image exceeds a certain threshold the visual feedback shall be paused to allow manipulation of the sand; were this not the case the users hands and arms would be considered as part of the sand surface and would be visualised. Temporal variance is susceptible to noise and may incorrectly classify slight shadows or projector flicker as motion. More robust motion detection methods like optical flow could be considered. This is similar to the approach taken by Kreylos et al. [4]. A small latency will be necessary to ensure that the system correctly registers the end of the manipulation, this will be achieved through a simple timer that will wait for a certain period of time before updating the visualisation, and will be tuned to ensure responsiveness.

3.5.9 RESTful API Remote User Interface

A key limitation of the existing AR sandbox solutions is the limited group control and engagement with the system as highlighted by Woods & Soltis et al. [5] [6], with the AR sandbox often limited to a single individuals use at any one time. This project aims to address this limitation by providing a RESTful API that allows users to interact with the system remotely. Kreylos et al. propose a similar approach but their implementation require direct application interfacing. This proposition differs in that the API shall be interfaced by a served website utilising WebGL and `THREE.js` to provide a 3D visualisation of the sandbox and simulation output alongside user controls inspired by the education tool *VisualPDE* by Walker et al. [38].

3.5.10 User Interactivness Testing & Evaluation Methodology

A series of unit tests will ensure that the individual components of the system are functioning correctly and that the data is being processed as expected. These tests will include checking the accuracy of the depth image processing, the calibration process, and the simulation output. Quantitative assessment of key metrics concerning latency will also be considered.

3.5.11 Deployment and Infrastructure

The system can be deployed on a single machine or in a distributed setup. In standalone mode, all parts run on the same workstation, suitable for small-scale educational use. With higher fidelity or larger domains, simulation tasks offloaded to a remote server running ExaHyPE in a Docker container. The client visualisation software sends elevation and boundary data, triggers the simulation, and receives the output via HTTP or socket-based communication in this case.

This deployment approach ensures consistency through containerisation, scalability via remote compute resources, and flexibility for adapting the system to various environments from education to research.

3.5.12 Benchmarking and Performance Evaluation

The performance of the visualisation system is evaluated across several key dimensions: frame rate, latency, resource utilisation, and rendering quality. Frame rate and latency are measured during real-time operation, with a target of 30 FPS for smooth interaction, taking into account the time from depth image acquisition to final projection including any simulation interfacing or user interactions. Additionally a resolution stress test will be conducted to evaluate the system's performance under varying conditions, including different resolutions and frame rates. Tests shall primarily be conducted under a standalone configuration as is common in existing literature, with the aim of providing a baseline for performance evaluation.

3.5.13 Development Methodology

The software and hardware components of the system are to be developed using an iterative and incremental approach, with a feature-driven development process. This approach allows for the rapid prototyping and testing of individual components, with a focus on delivering a working system as quickly as possible. Development will be undertaken by a single individual in contrast to the many projects found in literature that are often group efforts. The development process ordered according to the following phases:

- **Phase 1: Depth Sensing and Image Processing** - Development of the depth sensing and image processing pipeline, including the calibration process and noise reduction techniques.
- **Phase 3: Visualisation Pipeline** - Development of the visualisation pipeline, including the rendering of the topographical map and simulation output.
- **Phase 2: Simulation Integration** - Integration of the ExaHyPE engine into the system, including the configuration of the simulation input and output formats.
- **Phase 4: User Interaction and Motion Detection** - Implementation of user interaction and motion detection techniques to enable real-time feedback.
- **Phase 5: RESTful API and Remote User Interface** - Development of the RESTful API and remote user interface to enable group interaction with the system.

3.6 Overall Evaluation Methodology

Qualitative and Quantitative assessments will be made according to the methodology prescribed in their respective sections to form an overall evaluation of the system. Additionally the system will be assessed against the outlined list of requirements, objectives and deliverables already described and introduced.

4 Results, Discussion & Evaluation

4.1 Physical Setup Evaluation

Figure 5 demonstrates the final and complete physical setup of the AR sandbox, including the projector, depth camera, tripod, and sandbox. Overall, the setup proved sufficient for the requirements of the project. Below is a comprehensive assessment of the performance of each component:

4.1.1 Component Functionality & Limitations

4.1.1.1 Depth Camera: The Intel RealSense D435i camera performed well in capturing depth images, with a max depth resolution of 1280x720 pixels and a frame rate of 30 FPS. The camera's wide field of view (FOV) allowed for effective coverage of the sandbox area with the implemented calibration techniques. Additionally the camera's depth sensing capabilities demonstrated good performance in various lighting conditions, including when fully lit by the projector. Noise was observed but was effectively filtered out.

4.1.1.2 Sandbox / Container: A plastic under-bed storage container was used as the sandbox, providing pre-fabricated solution. The container had dimensions $79 \times 59 \times 18.5$cm providing a sufficient volume for sand manipulation; additionally a near $4:3$ ratio between width and length matched projector aspect ratio allow some degree of physical alignment.

4.1.1.3 Sand: Standard play sand was used, providing a fine texture for good manipulability; allowing it to be easily shaped and formed into various topographies and scenarios. The sand was also relatively inexpensive and widely available. 40kg of sand filled the container to a depth of approximately 12cm. Whilst this was sufficient for the project, an aquarium sand or similar white sand may have provided a more visually appealing surface for projection; which is further explored in the visualisation section.

4.1.1.4 Projector: A significant saving was made by selecting a low-cost projector, the *XuanPad Mini Projector 1080p* sourced from *Amazon*. Despite being a low-cost option, the projector performed sufficiently in terms of brightness and resolution. With a maximum resolution of 1920×1080 pixels, with both $16:9$& $4:3$ aspect ratios available, the projector was able to maximise the output of the visualisation ensuring full utilisation of the enhanced depth camera. Unfortunately the brightness did prove insufficient in bright ambient light conditions, with the projector unable to provide a clear image. This was mitigated by using the projector in a partially darkened room, but this may limit the usability of the system in certain environments.

4.1.1.5 Tripod & Camera Mounting: A multi-arm tripod was sourced to mount the depth camera and projector, providing an adjustable platform. Use of a pre-fabricated tripod was sufficient for the project, but a custom mount may have provided a more stable and secure platform for the camera and projector. Although an adjustable solution did allow for easy calibration of the system. Stability was a concern, with the tripod placed on a table to ensure that the projector was at a suitable height above the sandbox (due to the throw-distance required for a clear image), a fabricated solution such as that employed by UC Davies may have addressed this notwithstanding the advantage of adjustability.

4.1.2 Cost

TABLE 2: Cost Breakdown of Physical Setup

Component	Cost (£ GBP)
Depth Camera (RealSense D435i) *	~236
Projector	42
Tripod *	~60
Sandbox Container	23
Sand (40kg)	26
Total	**387**

** Items marked with an asterisk were internally sourced from Durham University Computer Science lab.*

The total cost of the physical setup was approximately £387 (GBP) (summarised in Table 2), with the depth camera being the most expensive component. Retail price reductions in home-use projectors have made them more accessible, with many low-cost options available allowing significant savings. Additionally the use of other pre-fabricated components such as the sandbox container and tripod allowed for a more cost-effective solution. Further the use of non-speciality sand (play sand) provided a low-cost solution; with some drawbacks. The total cost of the system is significantly lower than existing AR sandbox solutions, which can cost upwards of $1000 (USD). This demonstrates the potential for low-cost AR sandbox systems to be developed using off-the-shelf components. Higher-educational institutions with existing computer vision laboratories or equipment may make further savings by using existing equipment as done in this project, reducing costs enabling further outreach and opportunities.

4.1.3 Computational Resources

The system was developed and tested on a standard laptop with the following specifications:

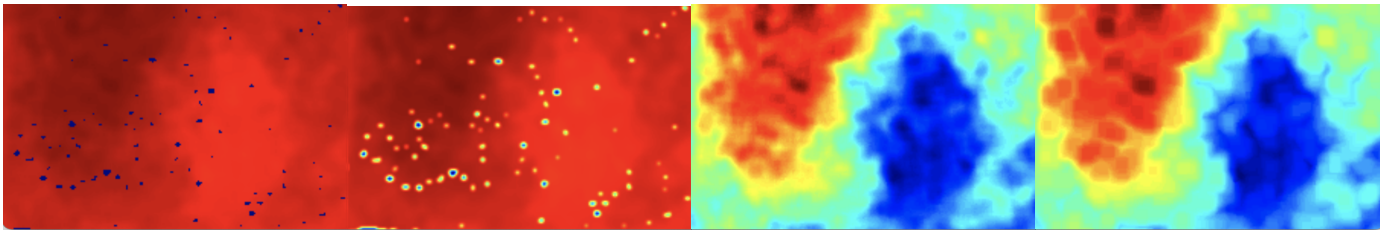
MacBook Pro 2019, Quad-Core Intel Core i5, 4x 1.4 GHz & Intel Iris Plus Graphics 645

These specifications allowed for sufficient performance during development and testing, enabling the full visualisation pipeline and output in real-time at a reasonable FPS. The system was somewhat burdened by the addition of the ExaHyPE simulation, although as previously noted this can be offloaded to a secondary machine, server or other computational resource.

4.2 Calibration & Projection Alignment

The calibration process was successful in aligning and centering the projected image within the sand surface. The calibration methodology involved the detecting the edges of the sand surface (ROI extraction), and then applying a perspective warp to align the projected image with the physical sandbox through the `OpenGL` texture coordinates.

Figure 6 illustrates the effectiveness of the ROI extraction process under low light conditions, with the sand edges distinctly outlined and masked in green. Additionally, the extreme points of the ROI are marked in red. Occasionally ROI process was ineffective, particularly in bright ambient or varying light conditions, causing the projected image to be slightly smaller than the defined region. However this was not a significant issue with the sand surface mostly covered.



(a) Original Depth Image (b) Gaussian Blur Applied (c) In-painting Applied (d) Morphological Dilation Applied

Fig. 11: Depth image processing results showing the effectiveness of various methods.

A homography matrix was then utilised to map these points into the projector space. This required the projection of a checkerboard pattern onto the sand surface, which was then captured by the colour camera. The corners of the checkerboard pattern were detected and used to compute the homography matrix, allowing for accurate mapping of the projected image to the physical sandbox.

Fig. 12: Calibration process showing the alignment of the projected checkerboard with the sand surface.

The application of `OpenCV` for ROI extraction and edge detection was successful, with the checkerboard pattern accurately identified. The computed homography matrix was then applied to warp the extreme points & projected image, achieving precise alignment with the physical sandbox, as shown in Figure 12.

Overall the full process delivered a robust and accurate calibration, able to ensure centering and alignment of the projected image within the physical sandbox where the projection space or sand surface was not perfectly physically aligned. This allows the physical setup to be somewhat ad-hoc and flexible, with the calibration process able to adapt to different setups and configurations; removing the need for a pre-fabricated aligned setup whilst also removing the manual calibration process found in literature.

4.3 Depth Processing & Temporal Filtering Performance

4.3.1 Data Cleaning & Preprocessing Results

To prevent simulation instability and visualisation artifacts an effective depth image processing pipeline resistant to variable lighting and projector interference was required. Several methods of CPU based image processing were explored, including Gaussian blur, in-painting, and morphological dilation and various RealSense provided and improvised approaches.

Since the RealSense camera occasionally produced clean depth images without imperfection (assessed by using a Jet colour map), a ground truth image was captured from a constant sand surface. Then the ambient lighting conditions were darkened with the projector remaining on, depth captures were then made and processed using the various methods. The depth image was then compared to the ground truth image using the SSIM metric.

Figure 11 shows the results of these experiments, Figure 11a contains the original depth image with noise that would lead to the aforementioned degradations highlighted by the jet colour scheme (dark blue regions).

Figure 11b illustrates the result of applying a 3×3 Gaussian blur to the depth image. The smoothing effect around the dark blue imperfections in the original image demonstrates that the Gaussian filter effectively reduces noise. However the dominant red areas of the normalised jet image indicate that noise and imperfections remain; demonstrating the need for a stronger kernel. Subsequently kernel sizes of 5×5 and 7×7 were applied, with the 7×7 kernel providing the best results. Although the image-wide application of the filter did degrade the overall depth image quality negatively translating into the simulation and visualisation with the loss of fine details.

TABLE 3: Structural Similarity Index (SSIM) for different preprocessing methods.

Metric	Gaussian Blur	In-painting	Dilation	RealSense
SSIM Value	0.84	0.96	0.94	0.93

In-painting was considered to eliminate the potential loss of fine details that could be observed with the Gaussian blur; seeing as it considers the missing pixels (holes) in the image rather than the complete image. In-painting was therefore applied to the original depth image, with a 3 pixel radius kernel; with its result shown in Figure 11c. The imperfections previously observed are now completely eliminated with a clear topography now distinct; emphasised by the jet map. This demonstrates the effectiveness of the in-painting function. However the computational expensive of the in-painting does frustrate the real-time performance of the system somewhat, making it less suitable in this case.

The morphological dilation method also produces a similar result (Figure 11d), effectively removing imperfections and revealing the topography. Notwithstanding its lesser SSIM score (see Table 3) compared to the in-painting method, it is computationally less expensive and still sufficiently effective for this application.

A markedly simpler yet effective solution is offered by the `librealsense` library, which provides a hole-filling filter that

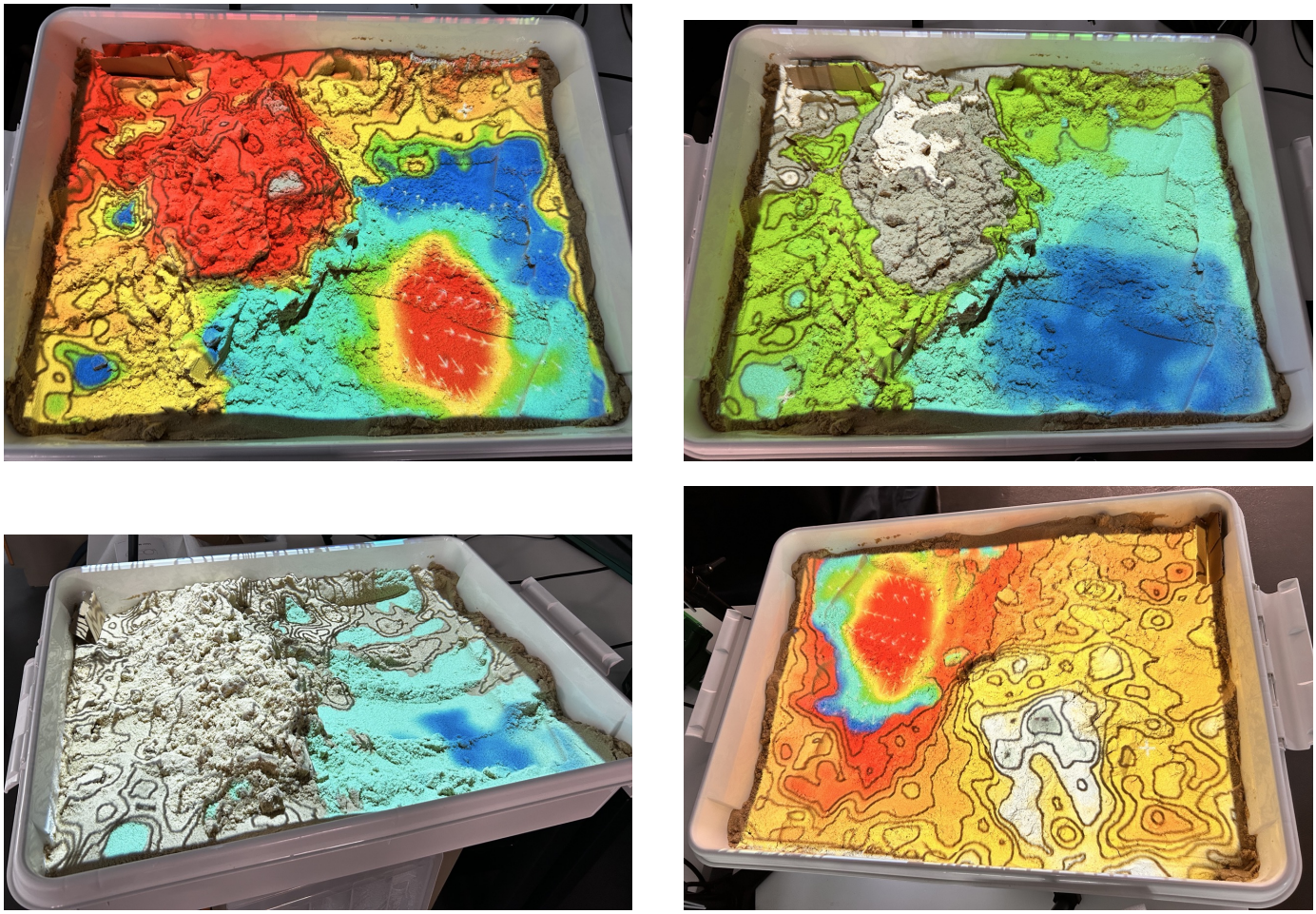


Fig. 13: Sample visualisation outputs showing various topographies, colour maps and scenarios.

can be applied internally to the image [39]. This filter considers a single pixel either its left, right or nearest neighbour and fills in the missing pixel with that value, rather than the neighbourhood averaging of a full dilation method. This method was ultimately selected for the final implementation due to its simplicity and effectiveness.

4.3.2 Temporal Filtering

TABLE 4: Comparison of temporal filtering methods.

Scenario	Mean (μ)	Std. Dev. (σ)	PSNR (dB)
Base	0.51	1.07	40.3
Dark RealSense	0.47	0.67	50.2
Lit RealSense	0.37	0.52	52.7
Low-pass (0.8)	0.48	0.65	49.1
5 Mov. Avg	0.45	0.49	54.3

Further noise is caused by temporal variance, which can be observed in the depth image as flickering and instability. This causes undesirable visual artifacts and noticeable flickering in the projected output. To address this issue, a temporal filtering method was implemented to smooth out the depth image over time. Three methods were evaluated under a variety of lighting conditions: a low-pass filter frame-to-frame filter, a moving average filter, and the *librealsense* provided method.

Of these the moving average filter was the most effective, with a significant reduction in flickering and instability, however a large window introduced a significant latency into the visualisation pipeline; hindering the real-time performance of the system and other components. The low-pass filter was also effective, but less so than the moving average filter. The `librealsense` filter, which uses a persistent cumulative averaging method, proved effective and computationally efficient, making it the preferred choice for the final implementation although the parameters α & δ were changed to ensure a level of stability suitable for the AR sandbox.

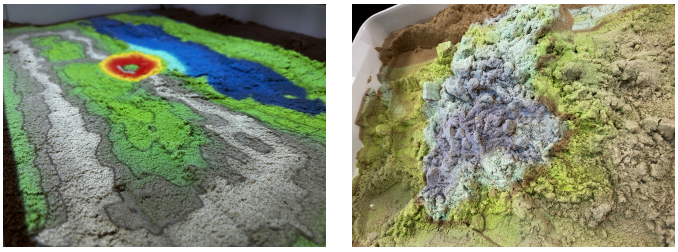
When combined with the motion filtering; only limited visual flickering in the projected output was observed with it being negligible in most cases.

4.4 Visualisation Performance

4.4.1 Topographical Output

The depth image was successfully passed into the visualisation pipeline, with a topographical map being constructed within an `OpenGL` shader and projected onto the sand surface. The topographical map was rendered in real-time, with the sand surface being updated as the user manipulated the sand when motion was absent. Figure 13 shows a series of images demonstrating the topographical output of the system, with various topographies and scenarios being visualised. The images show the effectiveness of the system in providing a clear and distinct visualisation of the sand surface, with the topography being accurately represented.

4.4.2 Ambient Light Conditions



(a) High ambient (b) Low ambient

Fig. 14: Ambient light conditions affecting the visibility of the projected image.

The use of a low-cost projector and ordinary play sand did limit the visualisation quality. This was primarily observed in high ambient light conditions, where the projected image was less visible and the sand surface was less distinct. Figure 14 shows the difference between high and low ambient light conditions, with the low ambient light condition providing a clearer image. This demonstrates the need for a more powerful projector or a darker environment to achieve optimal visualisation quality.

4.4.3 Colour Mapping

With colour mapping commonly used in topographical visualisation, it was important to ensure this core feature was implemented to provide clear, distinct and informative visualisations. Figure 13 shows a range of colour maps each providing its own distinct visualisation of the topography. Since colour maps are dynamically passed to the shader, users can select or customise the colour map to suit their preferences. This allows for a more engaging and interactive experience, with users able to select a colour map that best represents the topography.

4.4.4 Contouring

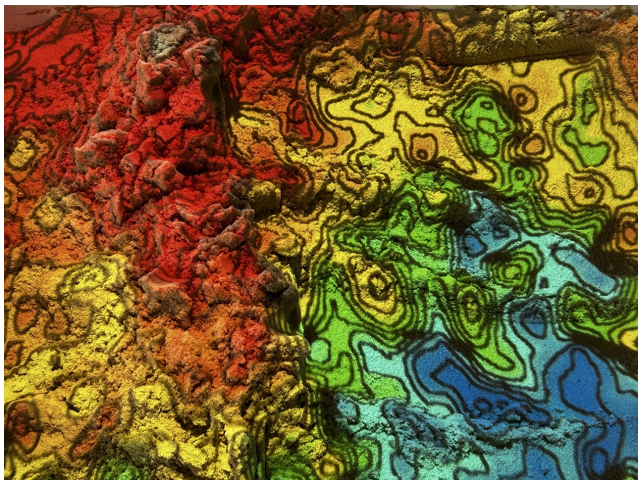


Fig. 15: Example of contour lines overlaid on a topographical map, illustrating elevation changes.

Another core feature of topographical visualisation is the use of contour / iso lines to represent elevation changes; these are essential to topographical visualisation so their implementation was carefully considered. The contour lines were generated using a simplified Marching Squares algorithm, which is a widely used method for generating contour lines from a heightmap. Once a user dynamically selects the contour interval, the algorithm is applied to the heightmap to generate the contour lines. The contour lines are then rendered in real-time within the `OpenGL` shader. Figure 15 demonstrates a dense application of contour lines to the sand surface indicating their effectiveness in providing a clear and distinct visualisation of the topography within the AR sandbox.

4.4.5 Water Simulation Texturing

Another component of the visualisation is the water simulation, which is visualised using a combination of colour mapping and texturing. The default water surface is rendered as a semi-transparent water blue-cyan surface, with the colour intensity representing the water height creating a depth effect. This allows for a clear and distinct visualisation of the water surface, with the colour intensity providing an indication of the water height.

Further informative visualisations are provided by the use of a vector field to represent the water flow direction and magnitude using the output data from the ExaHyPE engine. This can be displayed upon a jet colour map emphasising the water height, or on a white background with the vector field overlaid. The vector field provides insight into the discretisation of the simulation whilst visualising the water flow direction and magnitude that is useful in conveying key concepts surrounding hydrodynamic events.

4.4.6 Difference Map

Another aspect of this AR sandbox seeks to enable research-driven visualisation and simulations, with the aim of providing a platform for researchers to visualise and simulate various scenarios. To achieve this, a difference map was implemented to allow users to augment existing topographical maps into the simulation, providing a height difference maps and more complex pre-processed textured outputs. These are provided to the application using a simple image file.

Reinarz et al. in their demonstration of ExaHyPE's SWE capabilities [2] exhibit a simulation by Rannabauer et al. [40] of the 2011 Tōhoku earthquake and tsunami. Motivated by this scenario, a topographical heightmap of Upper Honshu and Hokkaido were created using `pyGMT` and passed into the application.

4.5 Wave Simulation Performance & ExaHyPE Integration

ExaHyPE was successfully integrated into the AR sandbox to enable research-grade hydrodynamic simulation, with its output visualised within the AR sandbox.

4.5.1 Tsunami Simulation and Wave Propagation

A primary motivation for integrating the ExaHyPE engine into the AR sandbox was to support more advanced hydrodynamic simulations, thereby creating a platform suitable for research-oriented scenario visualisation. Tsunami simulation encapsulates this objective and was subsequently investigated. Figure 17 illustrates the successful simulation of tsunami wave attenuation and propagation across the sandbox surface, demonstrating the system's ability to represent dynamic fluid behaviour over complex topography within the AR sandbox.

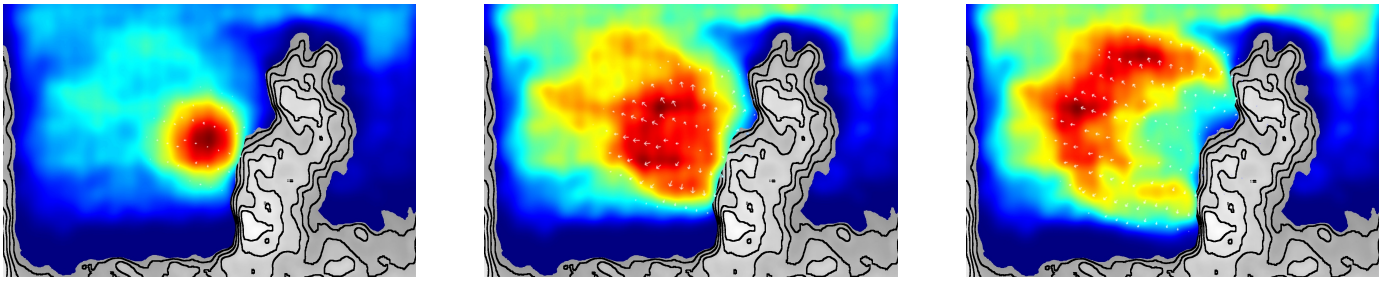


Fig. 16: Gaussian solitary wave rebounding off coast and resultant wave propagation.

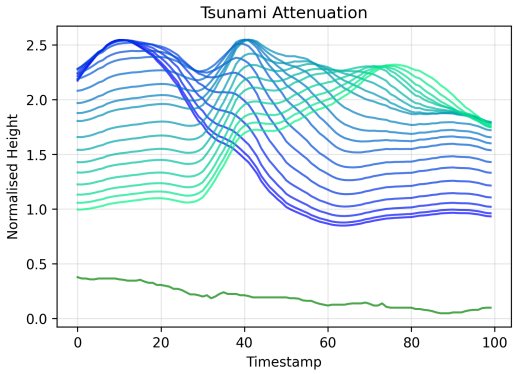


Fig. 17

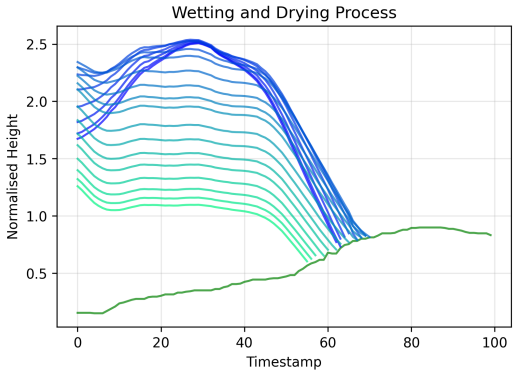


Fig. 18

Further experiments were conducted to examine wave propagation and overall behaviour at the terrain interface, since the fluid interaction with the terrain is a key aspect of tsunami simulation and the core concept of the AR sandbox. Figure 16 shows the results of a Gaussian solitary wave rebounding off the coast and propagating across the sandbox surface with an expected attenuation and dispersal of the wave; further highlighting the success of this integration.

4.5.2 Wetting and Drying

A realistic wetting and drying process was desired to enhance the realism of the simulation; informing the selection of the ExaHyPE engine. Partial success was achieved as demonstrated in Figure 18 examining the wet-dry boundary within a solitary wave scenario. It is observed that the water at $t = 0$ is at a mid height, it rises under the force of the wave and then recedes back and below its $t = 0$ position as the wave propagates and dissipates away from the terrain interface. From the figure it can be observed that this boundary is strictly enforced where a more realistic simulation would permit further wetting of the terrain. Further tuning of the ExaHyPE parameters may be required to achieve more realistic enforcement of boundary conditions at the interface, alternatively a more complex numerical scheme or control conditions may be required.

4.5.3 Stability and Performance

The model demonstrated overall stability, with only minor instabilities observed in specific scenarios. Such instabilities are to some extent inevitable, given the user-defined nature of the simulation domains and the inherent complexity of fluid dynamics. Nevertheless, the preceding figures confirm that the system is capable of producing high-quality and visually coherent simulations. Further tuning might alleviate some of these instabilities, but the current performance is satisfactory for the intermediate project goals.

4.6 Interactivity and User Responsiveness

4.6.1 Motion Detection

(a) Sand Manipulation (b) Motion Mask

Fig. 19: Low-pass temporal filter and thresholding for motion detection.

To maintain a balance between interactivity and visual clarity, a motion detection algorithm was implemented to identify user manipulation of the sand surface and temporarily pause the visualisation. This was accomplished using a basic low-pass filter combined with a thresholding mechanism to detect significant pixel differences in the depth image. A high threshold was employed to ensure that only substantial changes indicative of deliberate user interaction were registered. This approach enabled a responsive and uninterrupted user experience, with real-time visualisation resuming seamlessly once the surface was stable.

4.6.2 Web Remote Interface Evaluation

A RESTful API was implemented to allow users to interact with the system remotely. This was achieved using a simple web server that serves a static HTML page with a `THREE.js`

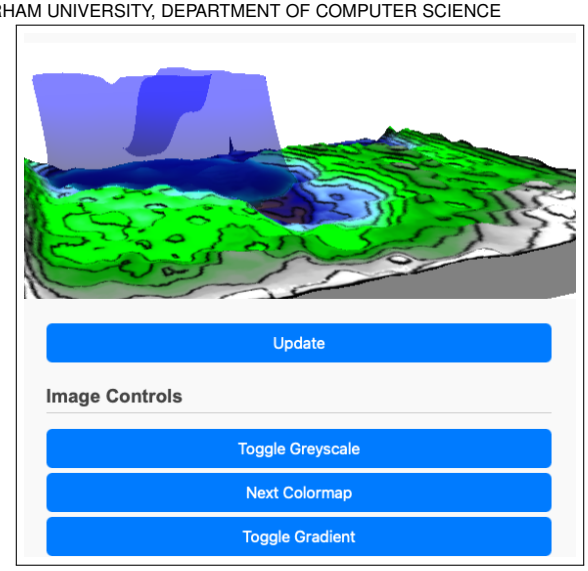


Fig. 20: `THREE.js` WebGL canvas with RESTful API.

enabled WebGL canvas along with accompanying JavaScript to process the interactions. This lightweight solution allows users to interact with the system using a web browser be that locally or on a mobile device sharing a network. HTTP requests facilitate the same interactions that are achieved locally using the keyboard, with the added benefit of allowing multiple users to interact with the system simultaneously. Users can view the sandbox within the page and interact with basic tools. This was only minimally explored in this project, but demonstrates the potential for a more interactive and engaging experience leveraging `WebGL` technologies.

4.7 System Integration and Overall Functionality

Overall the system was successfully integrated and functional, with all components working together to provide a cohesive and interactive experience. The depth camera was able to capture the sand surface and provide depth images to the visualisation pipeline, which was able to render the topographical map in real-time. The ExaHyPE engine then provided the hydrodynamic simulation, with the output being projected upon the sand surface.

4.8 Project Organisation and Management Evaluation

The organisation and management of this project were guided by a structured timeline and feature-driven development approach, ensuring the delivery of a functional AR sandbox system within the constraints of time and resources. The project was divided into distinct phases, including literature review, hardware purchasing & setup, software development, integration, and evaluation, with progress tracked using a Gantt chart. Task prioritisation focused on delivering core functionality early, enabling iterative testing and refinement, though some advanced features were de-prioritised due to time constraints. Unforeseen challenges, such as hardware compatibility issues and software integration complexities arose that required adjustments to the project plan.

As a single-author project, the scope was necessarily constrained compared to the team-based efforts observed in literature. However, this limitation was offset by the ability to maintain a cohesive vision and ensure consistent delivery of the project's measured objectives.

5 Conclusion

This project sought to re-examine the AR sandbox from a computer science perspective, upgrading many of the core concepts and components to provide a more robust and flexible system. Further by integrating the ExaHyPE engine, the project aimed to provide a research-grade hydrodynamic simulation platform suitable for educational outreach and research purposes. The project successfully met these goals, resulting in a functional AR sandbox system that embodies the intended improvements and capabilities.

The project demonstrated the feasibility of using automated calibration techniques to streamline setup and improve alignment precision, enabling the effective use of pre-fabricated components. This approach not only enhanced the system's accessibility and user-friendliness but also contributed to cost reductions further compounding this benefit. The integration of the Intel RealSense D435i camera further simplified coding practices and provided higher-resolution depth imaging, resulting in more accurate and detailed topographical maps and textured fluid simulations providing informative field maps with potential for expansion. Integrating the ExaHyPE engine into the AR sandbox allowed for advanced hydrodynamic simulations, including tsunami wave propagation and wetting-drying scenarios, showcasing the system's potential for interdisciplinary applications in education and research. Combined these features iterate and advance the existing success of the AR sandbox concept.

5.1 Future Work & Limitations

Whilst the project successfully achieved its objectives, there are several areas that for want of time and other constraints offer opportunities for potential future work and enhancements. These include further tuning of the ExaHyPE engine to improve the wetting-drying process, and overall stability. Whilst also considering the potential for real-time fluid simulation within ExaHyPE as opposed to the batch processing currently employed. This would allow for a more interactive experience, with users able to manipulate the sand surface and observe the effects in real-time. The concept of gesture control from literature could then be introduced to deliver a highly capable and fully interactive system. Additionally, the web interface could be further developed to enhance the group deployment potential building on the foundation explored.

The ground-up development of the project underscored the significant role of computer science in shaping the AR sandbox, as many foundational elements are rooted in core computer science & vision principles. This reinforces its value as an interdisciplinary outreach tool, capable of engaging students with computer science and engineering concepts through hands-on interaction extending its educational impact beyond its current use in geoscience.

Acknowledgements

I would like to express my gratitude to my supervisor Dr Anne Reinarz for her guidance and support throughout this project and for assistance with integrating ExaHyPE into my work.

I would further acknowledge the use of the Perception Systems (Heavy Duty) Lab at Durham University; providing access to the necessary cameras, equipment, space and resources to complete this project, with thanks extended to Professor Toby Breckon for facilitating this.

REFERENCES

[1] C. R. Amburn, N. L. Vey, M. W. Boyce, and J. R. Mize, "The Augmented REality Sandtable (ARES):," Defense Technical Information Center, Fort Belvoir, VA, Tech. Rep., Oct. 2015. [Online]. Available: http://www.dtic.mil/docs/citations/ADA622471

[2] A. Reinarz, D. E. Charrier, M. Bader, L. Bovard, M. Dumbser, K. Duru, F. Fambri, A.-A. Gabriel, J.-M. Gallard, S. Köppel, L. Krenz, L. Rannabauer, L. Rezzolla, P. Samfass, M. Tavelli, and T. Weinzierl, "ExaHyPE: An engine for parallel dynamically adaptive simulations of wave problems," *Computer Physics Communications*, vol. 254, Mar. 2020. [Online]. Available: https://durham-repository.worktribe.com/output/1275990/exahype-an-engine-for-parallel-dynamically-adaptive-simulations-of-wave-problems

[3] SMARTmania.cz, "Interactive Kinect Sandbox YouTube Video." [Online]. Available: https://www.youtube.com/watch?v=8p7YVqyudiE

[4] O. Kreylos and University of California, Davis W.M. Keck Center for Active Visualization in the Earth Sciences (KeckCAVES), "Augmented Reality Sandbox." [Online]. Available: https://web.cs.ucdavis.edu/~okreylos/ResDev/SARndbox/

[5] T. Woods, S. Reed, S. Hsi, J. Woods, and M. Woods, "Pilot Study Using the Augmented Reality Sandbox to Teach Topographic Maps and Surficial Processes in Introductory Geology Labs," *Journal of Geoscience Education*, vol. 64, pp. 199–214, Aug. 2016.

[6] N. A. Soltis, K. S. McNeal, R. M. Atkins, and L. C. Maudlin, "A novel approach to measuring student engagement while using an augmented reality sandbox," *Journal of Geography in Higher Education*, vol. 44, no. 4, pp. 512–531, Oct. 2020. [Online]. Available: https://doi.org/10.1080/03098265.2020.1771547

[7] T. Belle, "The augmented reality sandbox: An interdisciplinary mediation tool to foster Development."

[8] J. Cockrell and H. L. Petcovic, "Teaching topography using 3D printed terrain in an introductory earth science course: A pilot study," *Journal of Geoscience Education*, vol. 70, no. 1, pp. 2–12, Jan. 2022. [Online]. Available: https://doi.org/10.1080/10899995.2021.1927569

[9] T. Leinonen, B. , Jaana, V. , Henriikka, and N. and Sawhney, "Augmented reality sandboxes: Children's play and storytelling with mirror worlds," *Digital Creativity*, vol. 32, no. 1, pp. 38–55, Jan. 2021. [Online]. Available: https://doi.org/10.1080/14626268.2020.1868535

[10] H. A. Taylor, H. Burte, and K. T. Renshaw, "Connecting spatial thinking to STEM learning through visualizations," *Nature Reviews Psychology*, vol. 2, no. 10, pp. 637–653, Aug. 2023. [Online]. Available: https://www.nature.com/articles/s44159-023-00224-6

[11] F. Wellmann, S. Virgo, D. Escallon, M. de la Varga, A. Jüstel, F. M. Wagner, J. Kowalski, H. Zhao, R. Fehling, and Q. Chen, "Open AR-Sandbox: A haptic interface for geoscience education and outreach," *Geosphere*, vol. 18, no. 2, pp. 732–749, Feb. 2022. [Online]. Available: https://doi.org/10.1130/GES02455.1

[12] E. Smith and T. Ainsworth, "Expansion of the Augmented Reality Sandbox to Include Additional Water Sources for Guided Educational Experiences," 2022.

[13] Z. Zhou, "Immersive Computing for Coastal Engineering," Ph.D. dissertation, University of Southern California, Aug. 2022.

[14] S. Tavakkol and P. Lynett, "Celeris: A GPU-accelerated open source software with a Boussinesq-type wave solver for real-time interactive simulation and visualization," *Computer Physics Communications*, vol. 217, pp. 117–127, Aug. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0010465517300784

[15] S. Sánchez, L. Delgado, M. Gimeno-González, T. Martín García, F. Almaraz-Menendez, and C. Ruiz, "Augmented reality sandbox: A platform for educative experiences," Nov. 2016, pp. 599–602.

[16] S. Reed, S. Hsi, v, "Augmented Reality Turns a Sandbox into a Geoscience Lesson," Jul. 2016. [Online]. Available: http://eos.org/science-updates/augmented-reality-turns-a-sandbox-into-a-geoscience-lesson

[17] S. E. Reed, O. Kreylos, S. Hsi, L. H. Kellogg, G. Schladow, M. B. Yikilmaz, H. Segale, J. Silverman, S. Yalowitz, and E. Sato, "Shaping Watersheds Exhibit: An Interactive, Augmented Reality Sandbox for Advancing Earth Science Education," vol. 2014, pp. ED34A–01, Dec. 2014. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2014AGUFMED34A..01R

[18] R. Bärthele, S. Sauer, and J. Wilkening, "Exploring the potential of an Augmented Reality sandbox for geovisualization," *Proceedings of the ICA*, vol. 5, pp. 1–6, Aug. 2023. [Online]. Available: https://ica-proc.copernicus.org/articles/5/1/2023/ica-proc-5-1-2023.html

[19] "JBA Trust Augmented Reality (AR) Sandbox." [Online]. Available: https://www.jbatrust.org/about-the-jba-trust/how-we-help/physical-models/ar-sandbox/

[20] AR-Sandbox.eu, "Augmented Reality Sandbox - AR-Sandbox.eu." [Online]. Available: https://ar-sandbox.eu/

[21] O. Wasenmüller and D. Stricker, "Comparison of Kinect V1 and V2 Depth Images in Terms of Accuracy and Precision," Nov. 2016.

[22] Maurice, "Intel RealSense D435 Review – Maurice's Musings." [Online]. Available: https://www.calvert.ch/maurice/2018/06/12/intel-realsense-d435-review/

[23] "Depth Camera D435." [Online]. Available: https://www.intelrealsense.com/depth-camera-d435/

[24] artteknika, "Artteknika/Magic-Sand-for-RealSense," ART Teknika Inc., Aug. 2024. [Online]. Available: https://github.com/artteknika/Magic-Sand-for-RealSense

[25] O. Kreylos, "AR Sandbox Calibration Procedure," Aug. 2013. [Online]. Available: https://www.youtube.com/watch?v=V_-Qn7oEsn4

[26] "Navier-Stokes equation | Definition & Facts | Britannica," Sep. 2024. [Online]. Available: https://www.britannica.com/science/Navier-Stokes-equation

[27] "Shallow water equations," *Wikipedia*, May 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Shallow_water_equations&oldid=1225451839

[28] A. Kurganov and G. Petrova, "A Second-Order Well-Balanced Positivity Preserving Central-Upwind Scheme for the Saint-Venant System," *Communications in mathematical sciences*, vol. 5, Mar. 2007.

[29] J. Boussinesq, "Théorie des ondes et des remous qui se propagent le long d'un canal rectangulaire horizontal, en communiquant au liquide contenu dans ce canal des vitesses sensiblement pareilles de la surface au fond." *Journal de Mathématiques Pures et Appliquées*, pp. 55–108, 1872. [Online]. Available: https://eudml.org/doc/234248

[30] Various, "Boussinesq approximation (water waves)," *Wikipedia*, Oct. 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Boussinesq_approximation_(water_waves)&oldid=1253281855

[31] V. Michel-Dansac, C. Berthon, S. Clain, and F. Foucher, "A well-balanced scheme for the shallow-water equations with topography," *Computers & Mathematics with Applications*, vol. 72, no. 3, pp. 568–593, Aug. 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0898122116302838

[32] O. Kreylos, "New AR Sandbox in UC Davis DataLab," Nov. 2022. [Online]. Available: https://www.youtube.com/watch?v=kjmgLoEpKic

[33] J. Hohenadel, "Development of an augmented reality sand-box for the demonstration of tsunami modeling," 2018. [Online]. Available: https://mediatum.ub.tum.de/node?id=1488480

[34] G. Luetzenburg, A. Kroon, and A. A. Bjørk, "Evaluation of the Apple iPhone 12 Pro LiDAR for an Application in Geosciences," *Scientific Reports*, vol. 11, no. 1, p. 22221, Nov. 2021. [Online]. Available: https://www.nature.com/articles/s41598-021-01763-9

[35] "OpenCV." [Online]. Available: https://opencv.org/

[36] A. Reinarz, L. Seelinger, L. Rannabauer, M. Bader, P. Bastian, and R. Scheichl, "High performance uncertainty quantification with parallelized multilevel Markov chain Monte Carlo," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. St. Louis Missouri: ACM, Nov. 2021, pp. 1–15. [Online]. Available: https://dl.acm.org/doi/10.1145/3458817.3476150

[37] Intel®, "Tuning depth cameras for best performance." [Online]. Available: https://dev.intelrealsense.com/docs/tuning-depth-cameras-for-best-performance

[38] B. J. Walker, A. K. Townsend, A. K. Chudasama, and A. L. Krause, "VisualPDE: Rapid Interactive Simulations of Partial Differential Equations," *Bulletin of Mathematical Biology*, vol. 85, no. 11, p. 113, Oct. 2023. [Online]. Available: https://doi.org/10.1007/s11538-023-01218-4

[39] "Post-processing filters." [Online]. Available: https://dev.intelrealsense.com/docs/post-processing-filters

[40] L. Rannabauer, M. Dumbser, and M. Bader, "ADER-DG with a-posteriori finite-volume limiting to simulate tsunamis in a parallel adaptive mesh refinement framework," *Computers & Fluids*, vol. 173, pp. 299–306, Sep. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045793018300392