Position-based fluid simulation for robotic injection sealing of pavement cracks

Samuel D. Schaefer
Department of Engineering
University of Cambridge

Jie Xu
Department of Engineering
University of Cambridge

Damian Palin
Department of Engineering
University of Cambridge

jx322@cam.ac.uk

Abir Al-Tabbaa
Department of Engineering
University of Cambridge

Fumiya Iida
Department of Engineering
University of Cambridge

Abstract

Automated quality crack sealing can significantly benefit the maintenance of road pavements, but there is difficulty in depositing the correct volume of sealant material into the hidden crack space. A simulated model of the material flow within a crack space would allow the development of a predictive control scheme, such that the repair robot can apply suitable trajectories and operational parameters to accomplish neatly sealed surfaces. For the first time, the position-based fluid (PBF) method, computationally cheap and fast but approximate modelling of fluid flows, is studied for its feasibility for sealant flow simulation in the robotic injection crack sealing scenario. A Real-to-Sim experiment is performed, in which a PBF simulation of sealant in a virtual robotic crack sealing environment is mirrored from the physical lab setup. The fluid simulation is tuned to match the real-world dynamics through comparison with 132 simulation runs, varying the artificial viscosity parameters U (fluid-fluid viscous interaction) and D (fluid-wall viscous interaction). It was found that U had a varied three-stage influence on the simulation error while D's negative influence on the simulation error only effectively applied to fluids satisfying $1 < U < 100$. Through comparing the physical and virtual crack sealing results, the simulation was validated with an average fluid level error of 1.26 mm along a 3.1 mm wide, 16 mm deep and 80 mm long artificial crack, which shows the usefulness of the PBF method for robotic injection sealing. Besides, the approximation nature and computational power requirement of the PBF method are also discussed accordingly.

Keywords: Fluid simulation; position-based dynamics; crack sealing; robotic injection; pavement maintenance; Real-to-Sim; construction robotics

1 Introduction

For road pavements, it is desirable to perform frequent preventative maintenance, before faults escalate into dangerous damage on which repairs are more costly and less effective (Johnson, 2000; Fwa, 2005). Crack

sealing is the most important preventative measure (Highways Maintenance Efficiency Programme, 2012). Due to difficulties in depositing the correct volume of sealant, conventional crack sealing techniques such as overband sealing and inlaid sealing (British Board of Agrément, 2010; Standards for Highways, 2020) leave large areas of exposed materials, as shown in Figure 1. This ruins the pavement appearance and potentially causes vehicle accidents with increased surface slipperiness (Murphy, 2016; Hensley, 2017). A safer repair would employ precise material injection without overspill, leaving only a thin band (crack width) of exposed material. Figure 1 shows the two typical crack treatments alongside the improved sealing technique, material injection.

Material injection along the cracks can be tedious and time-consuming for human workers, yet robotic systems are particularly adept at performing such repetitive motions at speed, and additionally remove the safety risks of manual processes (Highways England, 2017). However, accurate planning and control of robotic crack sealing motions (e.g., nozzle trajectory, speed) with injection parameters (e.g., flow rate, nozzle diameter) can be a major challenge due to the hidden dynamics of sealant in the crack space. The fill level at any point along the crack is determined by the flow of the injected fluid; this flow depends not only on the robot's actions, but also on material viscosity and crack geometry. Current literature has not reported any effective methods to address this issue.

To uncover such hidden dynamics, numerical computer simulations are necessary. These can depict, understand and predict the behaviour of a given sealant fluid in a known crack space to inform the injection operation of sealant over time. Physics-based simulations (i.e., computational fluid dynamics) can give accurate predictions, but generally require high computational resources and therefore take a long time (Choi et al., 2021). Given the time requirement of the robotic sealing process, light-weight approximate simulations such as the Position-Based Fluids (PBF) method (Macklin and Müller, 2013) could be a preferable solution. The PBF method uses the Position-Based Dynamics framework, a fast and stable simulation technique directly acting on the vertex positions of object meshes rather than forces or impulses, used to model various dynamic systems such as rigid bodies, soft bodies and fluids (Tsai, 2017).

This paper examines for the first time the feasibility of applying PBF simulation for capturing the fluid behaviour in a robotic injection crack sealing scenario. The overall research approach is based on a Real-to-Sim experiment where a virtual robotic crack sealing environment is created, informed by the corresponding physical setup in the lab condition, and the PBF simulation method is used to model the crack sealing fluid. By varying the two artificial viscosity parameters for defining fluid-fluid and fluid-wall viscous interactions of the simulated materials (U and D respectively), a variety of fluid behaviours are achieved and compared with real-world dynamics. Such comparison is used to validate the effectiveness and potential of the PBF simulation for robotic injection sealing of pavement cracks.

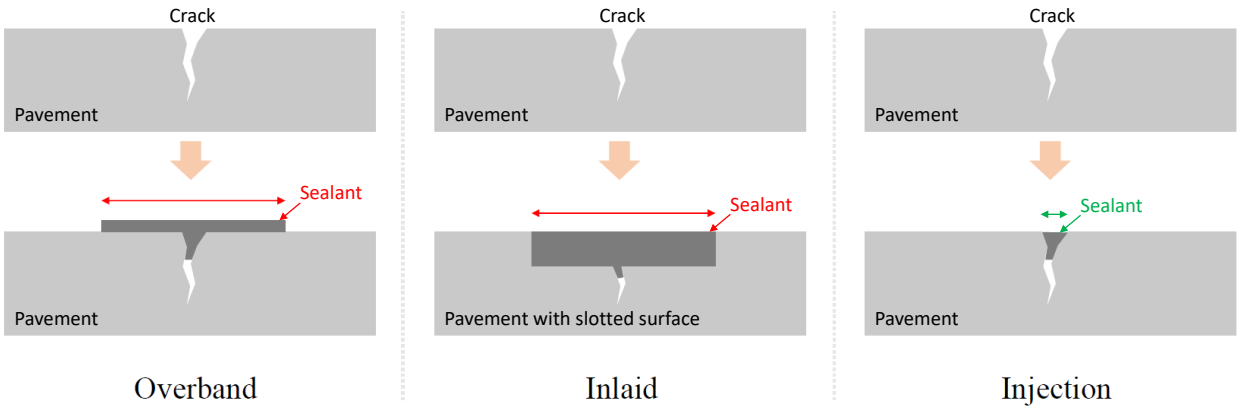


Figure 1: Three methods of pavement crack maintenance. Overband and inlaid sealing are current practices, while material injection could produce a safer road surface with less exposed sealant.

2 Related work

2.1 Automated crack sealing

There has been significant research in automated crack sealing for over 30 years, particularly for highway maintenance (Seo et al., 2014; Schaefer et al., In Preparation). These include the use of large trucks with robotic arms (e.g. (Velinsky, 1993; Pioneer Industrial Systems, 2022)), wheeled mobile robots (e.g. (Hong et al., 1997)), and trailers with XY gantries (e.g. (Haas et al., 1999; Lee et al., 2006)). The majority of such previous work has focused on utilising robotic systems for automated overband sealing (Haas, 1996; Haas et al., 1999; Lee et al., 2006; Holmes, 2010; Pioneer Industrial Systems, 2022), although there have been attempts at other techniques such as using a layer-by-layer 3D printing process to fill cracks (Torbaghan et al., 2019; Jackson et al., 2020; Awuah and Garcia-Hernández, 2022).

In a related area, overband sealing robots (Chen et al., 2022) and material injection robots (Yang et al., 2022) have been used to repair cracks in vertical concrete walls. Depending on the cracking severity, some of the repair materials used in these cases can be highly viscous to maintain their form in the cracks against gravity, so the filling process is more easily handled than that of pavement cracks where sealants are expected to settle and fill the cracks under gravity (Al-Qadi et al., 2006).

(Guo et al., 2017) and (Zhu et al., 2019) explored pavement crack injection sealing using a mobile robot and a 6-DOF robotic arm respectively. In both cases, the crack cross-sections were assumed to be constant, with hand-built demonstration cracks. Real-world pavement cracks generally have unknown and/or varying cross-sections, making the material dynamics more complicated, so repairs on real pavements would exhibit underfill and overspill.

(Liu et al., 2021) attempted injection sealing using an X-Y gantry with an asphalt extruder, adding a water-level sensor to cut off the flow when fluid reached the crack surface. Despite this, the process produced significant overspill, which was attributed to the water-level sensor being too far from the pavement, especially for uneven surfaces. The gantry was also found to be inaccurate, with drift in the extruder position over time. We argue that there is potentially a more significant issue: controlling the flow rate with a sensor gives it the ability to react to the environment in the current moment, but it cannot account for the future flow of liquid with the environment changing within the crack. Any lags within the controls, such as in moisture detection or nozzle shutoff, will cause overspill, while any segment of crack filled perfectly at one moment will be underfilled once the fluid has spread.

The shortcomings in prior work call for the use of a predictive model of the liquid sealant dynamics. Specifically, a fluid simulation model which is fast enough to run in a reasonable time, but also provides enough fidelity to read meaningful insights. Simulation is well suited to a robotic application, as the predictions can be plugged directly into the architecture to inform planning and/or real-time control.

2.2 Simulated liquids

To model fluids, traditional physics-based simulation involves two successive integrations: the first to calculate the velocity from forces acting on the elements, and the second to update the position based on the calculated velocity. For a complete description of physics-based computational fluid dynamics (CFD) methods, see (Hosain and Fdhila, 2015). One popular fluid simulator is Smoothed Particle Hydrodynamics (SPH), which models the fluid as a collection of particles. To enforce incompressibility, forces between neighbouring particles are calculated, such that the particles are repelled towards a rest density configuration (i.e., a state where particles are a specified distance from each other). The downside of this physics-based method is that a small time step is required for stability, so the simulation speed is limited.

To eliminate this limitation, Macklin and Müller (2017) developed a PBF simulation algorithm, using

position-based constraints to enforce incompressibility (Macklin and Müller, 2013). Neighbouring particles are incrementally moved such that they reach a configuration matching the rest density, changing the positions directly rather than calculating forces and integrating twice. This improves the stability of the simulation, allowing for far greater time steps, and therefore reducing simulation time. It also may improve robustness to modelling inaccuracies (Müller et al., 2007). However, this update step is not based on physical forces, but rather aims to give the appearance of fluid flow. Therefore, it has traditionally been reserved for computer animation and video games.

(Battaglia et al., 2013) suggests that approximate simulation techniques used in computer games can be useful as rough predictive models for robots. In fact, the speed and robustness of the PBF method make it ideal for use in robot decision-making (Guevara et al., 2017). In PBF simulations, material properties are decoupled from their analogous real-world properties (Guevara et al., 2017). This prohibits their use as a universally accurate predictive simulator. However, PBF simulations may have some predictive power, provided the environment remains roughly constant. That is, a simulator can be calibrated to a specific task and retain predictive power over the problem space.

(Pan et al., 2016) used a physics-based fluid dynamics simulation to optimise robot pouring actions. Fluids of different viscosities were simulated, although no real-world experiments were performed so the performance was not validated. Multiple iterations of simulating fluid flow for a given pouring action were performed, with the pouring action optimised until no spill was observed. During optimisation, assumptions about the fluid trajectory allowed gradient information to be easily obtained. Each iteration of the simulation and optimisation process took 54 minutes to complete, with tasks involving low viscosity liquids optimised in 1-2 iterations ($\sim$1-2 hours) and those with more viscous liquids requiring at least 6 iterations to achieve no spill ($\sim 5\frac{1}{2}$ hours). This would be prohibitively long for real-time liquid manipulation.

For a similar water-pouring task, (Guevara et al., 2017) used a PBF simulation to model the water, applying Bayesian optimisation to optimise the pouring action. The simulation was calibrated on real-world robotic water-pouring experiments, by minimising the difference in the volume of simulated water spilled and real-world water spilled. Simulations took only 20s to run, and iterating over more simulations improved the robot action, showing the value of the approximate PBF simulations in improving robot performance. The pouring container was constrained to rotate about an axis, and control was limited to two parameters, a constant angular speed and final angle, rather than allowing time-dependent pouring speed or unconstrained motion in the 3D space.

This research explores the implementation of a PBF simulation for a robot injection crack sealing scenario. The work also explores the effect of simulation parameters on the material's behaviour in the crack sealing task, by analysing fluid dynamics that would be unobservable without the aid of simulation.

3 Methods

To conduct the feasibility study, this research adopts a Real-to-Sim experimental approach. Firstly, a physical robotic injection crack sealing test with damaged pavement samples and a selected sealant is conducted in a laboratory. Then, a PBF simulation is adapted for the crack sealing process. Simulation of the robotic injection crack sealing process is explored through: 1) converting the physical elements into virtual elements; 2) particle modelling and PBF simulation implementation of the sealant injection processes; and 3) manipulating combinations of the two key parameters for defining fluid-fluid and fluid-wall viscous interactions to generate various behaviours of the simulated sealant; 4) analysing and comparing the simulated fluid behaviours with the real-world dynamics in the physical crack sealing test. Figure 2 depicts the overall methodological framework of the Real-to-Sim experimental approach.

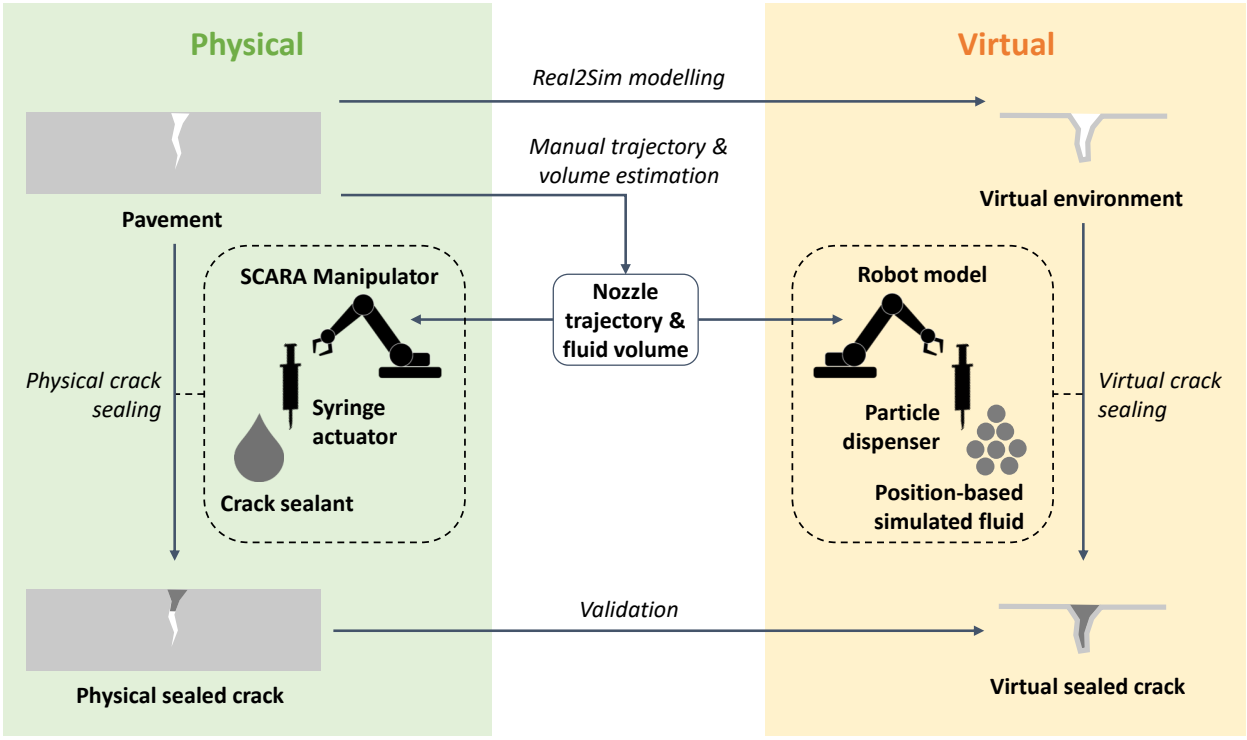


Figure 2: Methodological framework of the Real-to-Sim experiment of robotic injection crack sealing. In this work, manual processes bridge the gap between physical and virtual crack sealing processes.

3.1 Physical robotic injection crack sealing

A cementitious mortar sample measuring approximately 40x40x160 mm was split at a 60° angle using a compressive testing machine. The two halves were offset and secured together with tape to prepare an artificial crack with a measured average width of 3.1 mm. The sealant was a cement paste whose consistency was determined by a simple empirical criterion using a syringe with a nozzle diameter of 2.4 mm: neither too inviscid (the sealant would drip heavily) nor too viscous (the sealant would require significant force to dispense and dry very quickly in the nozzle). Preliminary experiments showed that a water-to-cement ratio of 0.67 produced a suitable consistency for the cement paste. The crack was partially filled by hand to create an average depth of 16 mm (manually measured with a ruler at multiple points along the crack length), reflecting real pavement cracks while reducing the volume of sealant needed and then the simulation time.

The nozzle trajectory was defined by digitally tracing the crack centreline on a top-down image, starting 10 mm from one end of the crack and ending 10 mm from the other. From preliminary experiments with similar cracks, a sealant volume of 4.8 ml was chosen. The nozzle travel speed was set to 4 mm/s for this experiment, so with the trajectory length of 60 mm, the crack sealing took 15 s to complete.

An Epson Scara T6 4-DOF robotic arm was fitted with a custom cement paste-injecting end-effector which consists of two 3D printed pieces and the syringe. The robot was programmed with Epson RC+ 7.0 to follow the list of trajectory points. For each point, the Z-coordinate of the robot, which controls the position of the plunger, moved a distance proportional to the XY distance between each waypoint. This ensured a constant flow rate of sealant into the crack, defined by the total nozzle trajectory length, volume of sealant, and nozzle travel speed, such that all the sealant was deposited when the nozzle reached the end of the path.

The prepared crack sample was sealed and left in the lab until the sealant had fully hardened. To quantify the quality of the physical crack sealing for the comparison with that of the following virtual crack sealing,

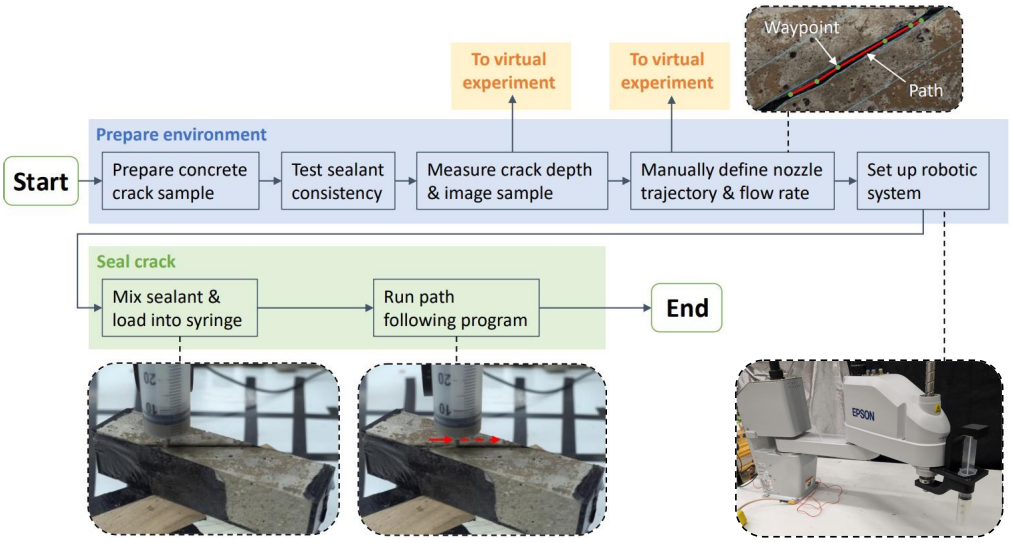


Figure 3: Physical robotic injection crack sealing flowchart.

the depth from the sample's surface to the sealant height level was measured every 5 mm along the crack using a pair of digital callipers, with the process repeated 3 times and an average measurement taken.

3.2 Simulation of robotic injection crack sealing

3.2.1 Converting physical sample to virtual sample

A virtual model of the concrete crack sample was created in Fusion 360 by tracing the sides of the crack from a top-down image of the sample, and extruding the resulting profile by 16 mm (the measured crack depth) to create a 3D surface mesh. At runtime, the surface mesh defining the crack geometry is converted to a set of stationary particles as follows. First, the vertices and normals are iterated over and transformed into world coordinates, to account for the position and orientation of the sample. The mesh model defines the outer edge of the wall, so the vertices must be offset to account for the radius of each particle along the direction normal to the facet. Each facet of the resulting mesh is subdivided to obtain an approximately uniform distribution of particles over the model.

With perfectly vertical walls, the fluid particles tend to either build up at the top of the crack, or fall through the crack, gaining enough velocity to pass through the floor in a single time step (i.e. the particle moves from above the floor to below the floor in a single time step, with no possibility of interaction with the floor particles). These scenarios are not observed in real life, where instead the deposited fluid hits the wall and flows down. Therefore, the crack walls were angled by 11°, which was found through preliminary testing to successfully funnel particles into the crack while keeping the appearance of the original crack. The process of converting an image of the sample into a virtual model is shown in Figure 4.

3.2.2 Position-based fluid algorithm development

The PBF simulation described in Algorithm 1 is based on (Macklin and Müller, 2013). The algorithm represents each fluid particle by its velocity $\mathbf{v}_i$ and centre position $\mathbf{x}_i$. In each time step Δt, the velocity is updated by gravity and damping, the particles move due to their velocities, and the positions are adjusted to satisfy an incompressiblity constraint. The new velocity is calculated by comparing the updated positions $\mathbf{x}_i^*$ with those of the previous time step $\mathbf{x}_i$. Details of the algorithm are provided below.

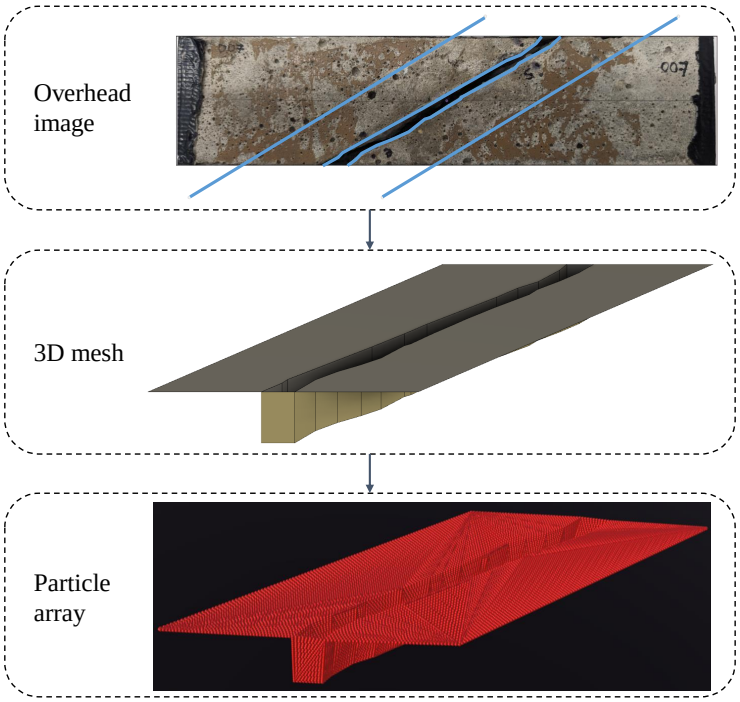


Figure 4: Generating virtual crack model for the overhead image of the physical sample and measured crack depth.

Algorithm 1 Position-based fluid

for all particles i **do**
  apply forces $\mathbf{v}_i \Leftarrow \mathbf{v}_i + \frac{\mathbf{f}_{ext}(\mathbf{v}_i)}{m}\Delta t$
  predict position $\mathbf{x}_i^* \Leftarrow \mathbf{x}_i + \mathbf{v}_i \Delta t$
end for
for all particles i **do**
  find neighbouring particles $N_i(\mathbf{x}_i^*)$
end for
while $iter < solverIterations$ **do**
  for all particles i **do**
    calculate λ_i
  end for
  for all particles i **do**
    calculate $\Delta \mathbf{x}_i$
    adjust positions $\mathbf{x}_i^* \Leftarrow \mathbf{x}_i^* + \Delta \mathbf{x}_i$
  end for
for all particles i **do**
  update velocity $\mathbf{v}_i \Leftarrow \frac{1}{\Delta t}(\mathbf{x}_i^* - \mathbf{x}_i)$
  while $iter < viscosityIterations$ **do**
    apply XSPH viscosity
  update position $\mathbf{x}_i \Leftarrow \mathbf{x}_i^*$
end for

In each time step, the particles are moved towards a configuration that satisfies the density constraint $\rho_i = \rho_0$, where ρ_i is the density at the ith particle and ρ_0 is the rest density (set to 1000 kg/m3). The

constraint is formulated as $C_i(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \frac{\rho_i}{\rho_0} - 1$, where we desire $C_i(\mathbf{x}) \to 0$. This is achieved by moving each particle along the gradient of its constraint with a scaling factor λ_i, so $\Delta\mathbf{x}_i = \nabla C_i(\mathbf{x})\lambda_i$. This scaling factor λ_i is a negative value (stepping towards $C_i(\mathbf{x}) = 0$), found by rearranging the first-order expansion of $C_i(\mathbf{x} + \Delta\mathbf{x}) = 0$:

$$\lambda_i = -\frac{C_i(\mathbf{x}_1, \ldots, \mathbf{x}_n)}{\Sigma_k |\nabla_{\mathbf{x}_k} C_i|^2 + \varepsilon} \tag{1}$$

The constant ε is required to prevent instabilities where particles are separating: as the distance between particles i and j becomes large, the gradient of C_i with respect to particle j converges to zero, which would ordinarily cause λ_i to become very large. The particle i would then move a great distance back towards particle j, overshooting and thereby causing a large move back in the opposite direction. Therefore, ε prevents λ_i from becoming too large. Here, $\varepsilon = 60$.

The gradient of the density constraint on particle i with respect to particle k, $\nabla_{\mathbf{x}_k} C_i$, is given by the popular physics-based fluid simulation technique, smoothed particle hydrodynamics. This models the effect of each constraint on neighbouring particles as a smooth function $W(\mathbf{x}_i - \mathbf{x}_j, h) = \frac{1}{h\sqrt{\pi}} e^{-(|\mathbf{x}|^2/h^2)}$, with closer particles producing a larger effect and particles far away producing no effect. This allows the gradient of the constraint to be written simply as the sum of gradients of that smoothing function over all particles j within a distance h of particle i:

$$\nabla_{\mathbf{x}_k} C_i = \frac{1}{\rho_0} \Sigma_j \nabla_{\mathbf{x}_k} W(\mathbf{x}_i - \mathbf{x}_j, h) \tag{2}$$

Each particle's density constraint has a reciprocal effect on neighbouring particles, so each particle is moved by:

$$\Delta\mathbf{x}_i = \frac{m}{\rho_0} \Sigma_j (\lambda_i + \lambda_j) \nabla W(\mathbf{x}_i - \mathbf{x}_j, h) \tag{3}$$

Particle corrections due to each constraint are calculated independently, without moving the particles. The new configuration is not guaranteed to satisfy the density constraint of particle i, as neighbouring particles will also move; rather it is a step towards satisfying the constraints. Therefore, a number of iterations of this position update step are performed until the position changes mostly converge to zero. In this work, $solverIterations = 5$.

As in (Macklin and Müller, 2013), external forces are gravity and damping. Damping is necessary to allow the system to settle to a stable configuration, and corrects for the possibility of position-based incompressibility constraints adding energy to the system. Therefore, the resultant force for each particle is given by:

$$\mathbf{f}_{ext}(\mathbf{v}_i) = m\mathbf{g} - \eta\mathbf{v}_i \tag{4}$$

where $\mathbf{g}$ is the acceleration due to gravity usually given by $\mathbf{g} = [0, -9.81, 0]^\intercal$, and η is the damping. A small, positive value of damping must be present to avoid instabilities due to energy addition in the incompressibility update. Preliminary experiments showed the exact value of η had no significant impact on material behaviour, so $\eta = 0.2$ was chosen.

Unlike the original algorithm, the simulation must be able to model a range of fluids, from inviscid to highly-viscous. High viscosities can cause instabilities in the viscosity update step. To resolve this instability,

the solver performs multiple passes with a lower viscosity (Alduán et al., 2017). This has less impact on performance than simply reducing the time step, as the time between incompressibility updates remains long.

The standard PBF algorithm neglects viscous interactions with the boundaries. Takahashi and Fujishiro provide an algorithm including wall drag (Takahashi and Fujishiro, 2013). The algorithm used here defines the walls by a set of stationary particles (fixed in space) and makes the additional assumption that the number density of wall particles is constant in the incompressibility update and viscosity update of fluid particles.

These two modifications lead to the modified artificial viscosity update (modified XSPH) calculation:

$$\mathbf{v}_i \Leftarrow \mathbf{v}_i - \frac{U}{q}\Sigma_j \frac{m}{\rho_j} W(\mathbf{x}_i^* - \mathbf{x}_j^*, h)\,(\mathbf{v}_i - \mathbf{v}_j) - \frac{D}{q}\Sigma_k W(\mathbf{x}_i^* - \mathbf{x}_k, h)\mathbf{v}_i \tag{5}$$

`Repeat` q `times`

where j denotes neighbouring fluid particles to particle i, k denotes neighbouring wall particles to particle i, q is the number of XSPH update steps chosen to maintain stability, U is the viscosity, and D is the wall drag. The traditional Greek letter μ is avoided in this description to avoid confusion with the equivalent parameters in the physical equations of motion; U (unitless) and D (length units, so that D/h forms a unitless quantity) are parameters determining the weighting given to the two viscous terms. The material behaviour can be altered by varying the parameters U and D.

3.2.3 Simulation implementation

Simulations are performed on an Intel Core i5 CPU with 8.00 GB of RAM and an Intel HD Graphics 520 graphics card. It runs in Unity 2020.3.11f1 (Unity Technologies, 2021) using GPU computation. The structure of a single simulation run is shown in Figure 5.

Initialise simulation: The 3D model of the crack geometry is converted into a set of stationary particles as described in Section 3.2.1. The nozzle path is imported from a *txt* file. The nozzle is positioned at the first waypoint, and the target point is set as the second waypoint. The nozzle speed is calculated such that the path is completed in the specified time, and direction determined by normalising the vector between first waypoint and target point. The number of fluid particles needed is calculated from the volume of sealant required using the formula:

$$N = \frac{V}{4f^3R^3\sqrt{2}} \tag{6}$$

where V is the sealant volume in mm^3, R is the particle radius in mm, f is a constant controlled by how closely the particles are packed (so the effective radius is fR). This is easily obtained from the close-packing sphere equation $\rho = \frac{\pi}{3\sqrt{2}}$. A value of $f = 0.97$ was found from preliminary experiments where particles were deposited into a cuboid of known volume.

Like the wall particles, any particle can be excluded from calculations by designating it as a stationary particle in the same way. Initially, all fluid particles are stationary, stored with arbitrary positions below the work surface. This is necessary to ensure the fluid solver receives position and velocity vectors with constant lengths.

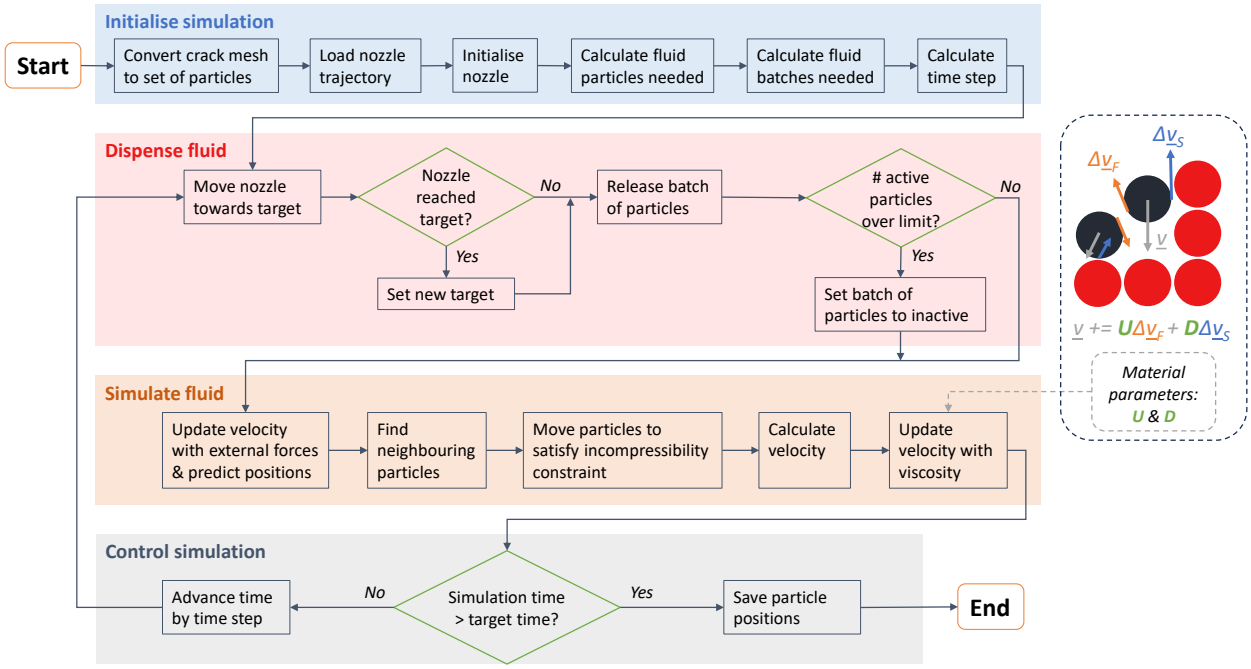


Figure 5: Robotic injection crack sealing simulation flowchart. Dashed box shows illustration of the viscosity update with parameters U and D, for fluid particles (black) and wall particles (red).

The diameter of the fluid particles was chosen based on previous simulation experience: if there are too few particles, the wall drag will easily dominate the viscosity updates on the particles. Therefore, to give sufficient freedom to the fluid behaviour, it was chosen to have 4 particles along the crack width, i.e., particle diameter $= 3.1$ mm$/4 \approx 0.8$ mm.

The number of fluid particles per batch is calculated from the particle diameter (0.8 mm) and the nozzle diameter (2.4 mm). The number of batches to release all the fluid is then determined, from which the time step is calculated such that one batch is released per frame for the specified time.

Dispense fluid: The nozzle is moved towards the target by first-order integration of the nozzle velocity. If the nozzle has reached the target point, the next target is set as the next waypoint in the list of trajectory points. A batch of fluid particles is set to active, with their positions set to a circular formation around the current nozzle position. Particles added early on in the process tend to become stationary as the simulation progresses. To reduce computation, the particles active for most frames are converted into stationary particles such that only 2,500 particles are active simultaneously.

Simulate fluid: The PBF implementation is based on the work of Justin Hawkins (Hawkins, 2008), with modifications to the viscosity solver explained in Section 3.2.2 (referring to Equation 5). The solver follows the steps in Algorithm 1. To exhaustively simulate fluids with different viscosities, the artificial viscosity parameters U and D are continuously altered to complete multiple runs until the best fit fluid behaviour is found. The illustration in Figure 5 shows the effect of these parameters: the velocity of each particle, $\mathbf{v}$, is updated to align with adjacent fluid particle velocities by adding $\Delta\mathbf{v}_F$, and slowed by the wall by adding $\Delta\mathbf{v}_S$; the parameters U and D control the strength of each term respectively. The full details can be found in Equation 5.

Control simulation: Screenshots are taken at time points 0.5 s, 7 s, and 15 s. If the simulation time has not reached the specified time of 15 s, to match the physical experiment, the simulation time is advanced by one time step, and the *dispense particles, simulate fluid* loop is repeated. If the simulation time is over the

specified time, the particle positions are saved to disk as a *.csv* file, and the simulation run ends.

3.2.4 Parameter manipulation and data collection

The artificial viscosity parameters U and D determine the strength of fluid-fluid and fluid-wall viscous interactions respectively. In rough terms, a higher U gives a more viscous liquid and a higher D gives a more adhesive liquid. This experiment seeks to analyse the behaviour of different simulated materials by running multiple simulations with the same environment but various combinations of U and D. Therefore, the script was run with 132 combinations of $10^{-2} \leq U \leq 10^{4}$ and 10^{-4} cm $\leq D \leq 3$ cm. Figure 6 shows cross-section views at several time-points from several simulations run with different viscosity parameters. The more viscous material forms higher peaks earlier, and spreads slower along the crack floor. The jet also spreads out more due to the greater interaction with boundary particles. The final particle positions are then analysed for each simulated material.

In addition, a Python script is used to obtain a fluid surface level contour in the following steps. First, the positions are used to fill an array representing the cross-section view in pixels. Next, smoothing is applied by convolution with a kernel of all 1s to account for the diameter of fluid particles. Each column of the resulting binary array is searched, and only the longest continuous line of fluid pixels is kept (this excludes erroneous overspill particles). Finally, the fluid surface level is obtained as the top-most pixels of the resulting columns.

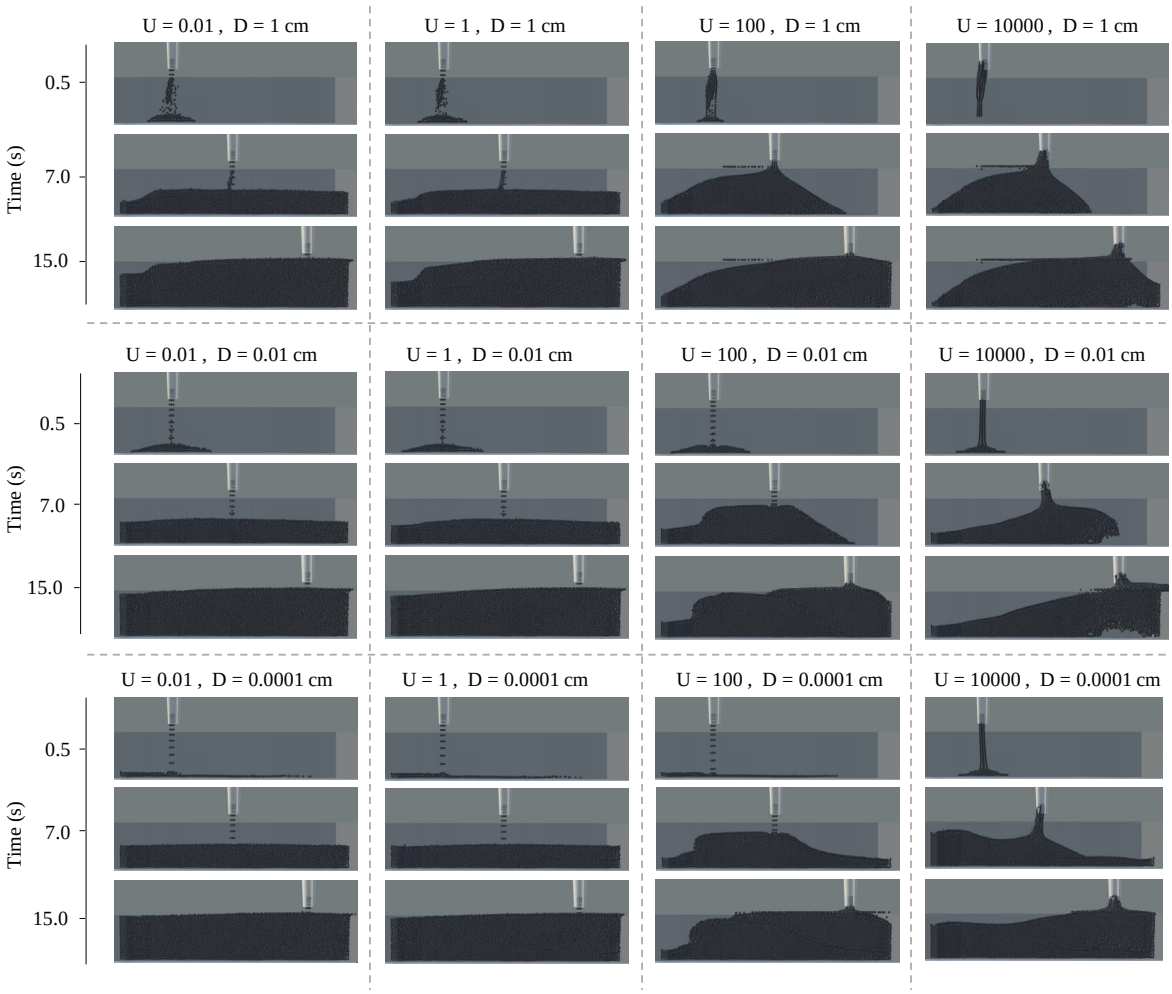


Figure 6: Cross-section views showing frames from simulations with varying viscosity parameters U and D.

4 Results

Figure 7 shows the final surface level for 132 simulated materials, along with the observed physical result from the robotic crack sealing trial. Several of these simulations show a good quantitative match between virtual and physical crack sealing events: significant underfill is observed at the left side of the crack; the level rapidly increases to around -4 mm at $x \approx 20$ mm, where it plateaus until $x \approx 30$ mm before another increase is observed; at $x \approx 40$ mm, the fluid level plateaues again, giving an area of overspill until the end of the crack around 1 mm high.

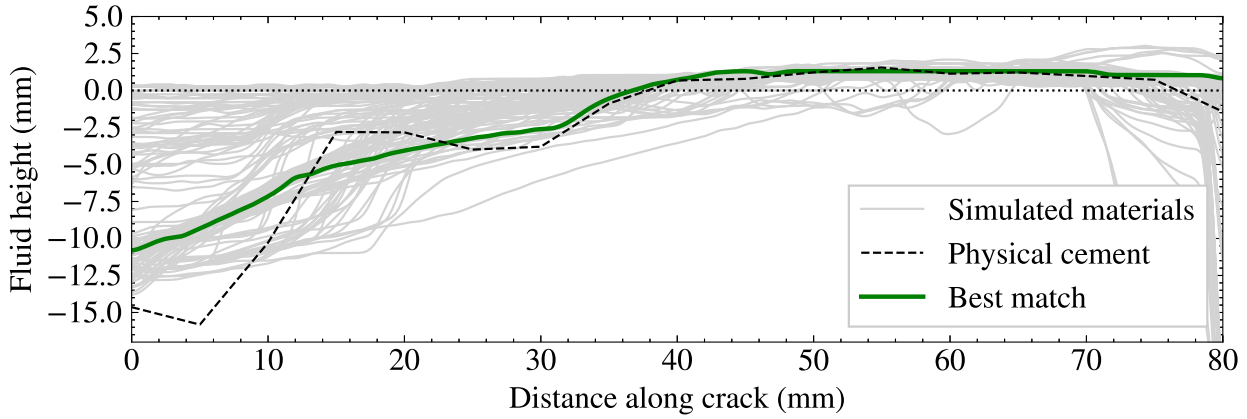


Figure 7: Final fluid level relative to the crack top surface, for different simulated materials (solid grey lines) compared with the physically injected sealant (dashed line); the best match simulation result is highlighted (bold green line).

To compare the simulations with the physical crack sealing trial, the average error between the simulated and physical final fluid level along the length of the crack was calculated. The area enclosed by each simulated final fluid level curve and the physical final fluid level curve was calculated using the *similaritymeasures* Python library (Jekel et al., 2019), and divided by the crack length (80 mm) to give the average error of the simulation. More accurate simulations minimise this error, so a perfect value in theory would be 0. Among all 132 simulations, the highest error was 3.45 mm, found with the most inviscid material ($U = 0.03$, $D = 0.0001$ cm) and the closest match achieved an error of 1.24 mm, highlighted by the bold green curve in Figure 7 ($U = 30$, $D = 3$ cm).

To allow global visualisation of the influences from U and D on the simulation accuracy, the average error of each simulation run is synthesised into a heat map as shown in Figure 8, with more accurate simulations in purple and less accurate simulations in yellow, which has shown two regions of higher accuracy (purple) and one region of lower accuracy (yellow-to-green). The full heat map is obtained with SciPy's *griddata* function using piecewise linear interpolation (Virtanen et al., 2020).

The larger purple region contains the parameter set with the highest accuracy: $U = 30$, $D = 3$ cm (error = 1.24 mm), highlighted by the bold green cross. However, higher values of D require a large number of viscosity iterations (q in Equation 4), and the simulation slowed significantly when $D > 1$ cm. Taking this into account, the simulated material with parameters $U = 30$ and $D = 0.3$ cm (error = 1.26 mm) is chosen as the most suitable, highlighted by the bold orange cross. The large purple region of high accuracy around the chosen artificial viscosity parameters also indicates that these parameters are robust to perturbations.

Comparison between physical and virtual crack sealing using the selected artificial viscosity parameters (U = 30, D = 0.3 cm) is provided in Figure 9. Viewing the scene status of physical and virtual crack sealing side-by-side at t = 10 s and t = 15 s in Figure 9(a), similar areas of underfill and overspill can be observed, although the simulated fluid overspill area (particles resting on the crack's top surface) shows slight discrepancy to the physical result, which visually validates the effectiveness of the applied PBF simulation method.

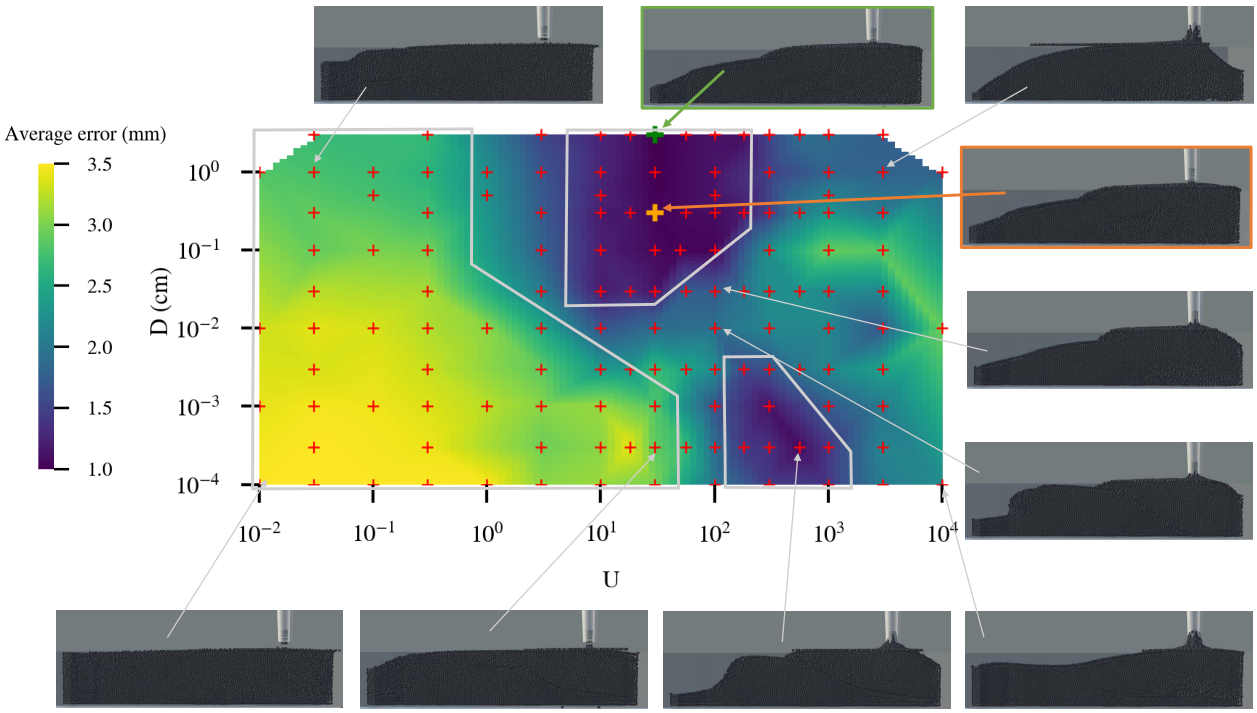
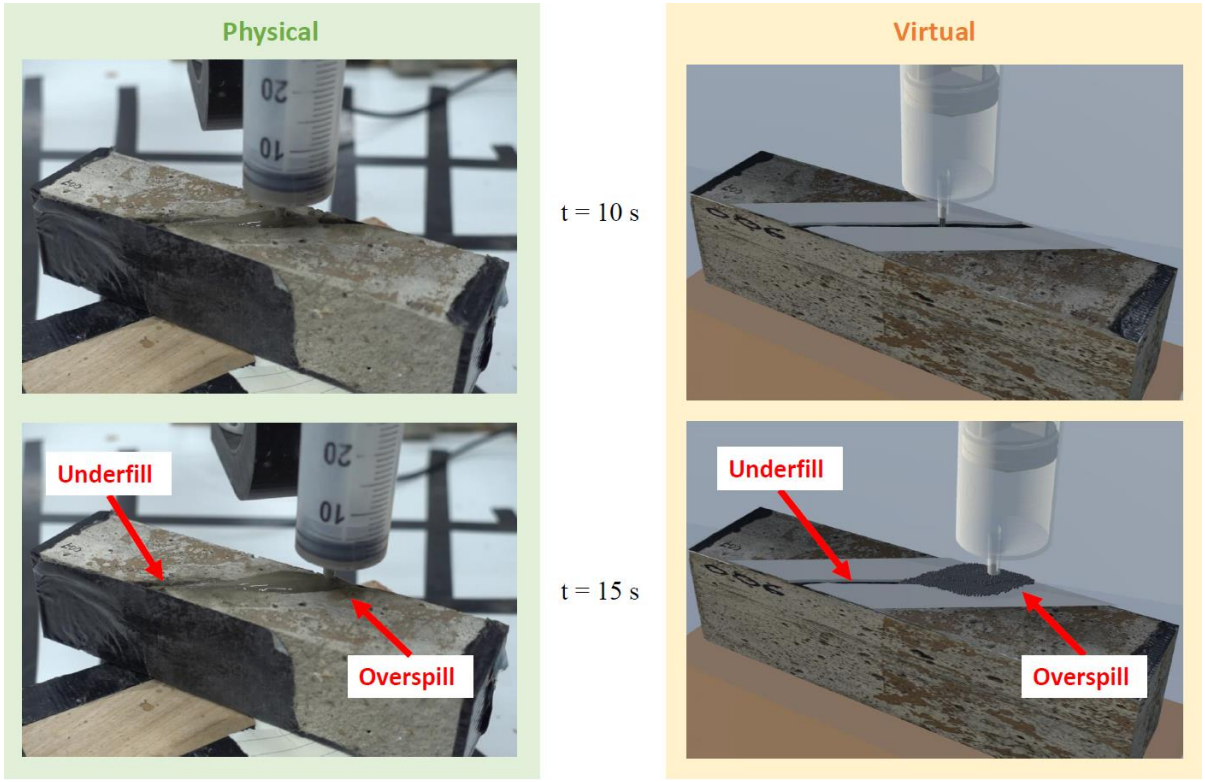
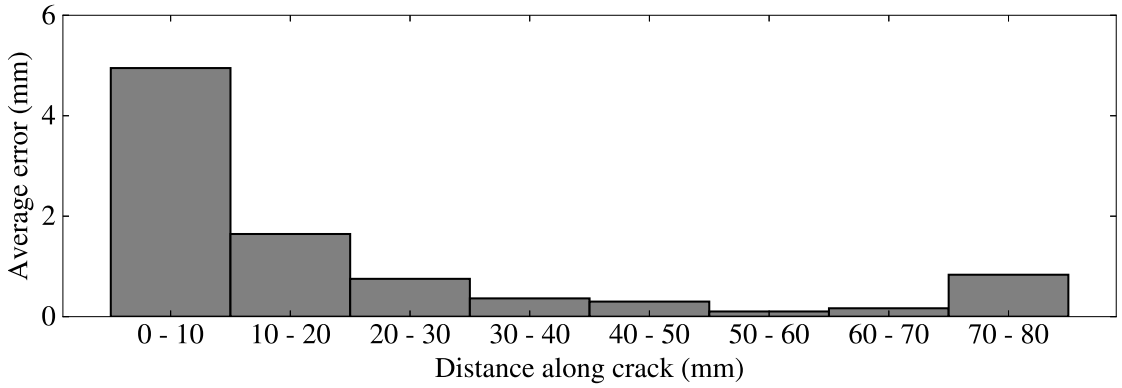


Figure 8: Heat map of the similarity score (simulation accuracy) on the final fluid profiles, for various values of artificial viscosity parameters U and D. Overlaid red crosses show the simulation trial parameters. Selected simulation runs are annotated with an image of the resulting fluid profile at 15 s.



(a)



(b)

Figure 9: Comparison between physical and virtual crack sealing using the selected artificial viscosity parameters ($U = 30$, $D = 0.3$ cm): (a) Scene status at t = 10 s and t = 15 s; and (b) the simulation error distribution along the crack length.

With the average simulation error of 1.26 mm along the whole 80 mm length of the crack, Figure 9(b) uses a histogram to present the distribution of this simulation error over the crack length. Starting from one end (zero distance), the simulation error drops immediately once passing the first 10 mm range of the crack length. The majority (20 - 80 mm range, 75% crack length) of the crack sealing is simulated at a level of less than 1 mm error, with a minimal error of approximately 0.1 mm in the 50 - 60 mm range, which is sufficiently good for use in a robotic control system. The maximal error can reach approx. 5 mm, but this only happens at/near the start location, which is reasonably expected as the physical sealing operation could be unstable at the start/stop locations. This also implies that additional care should be taken around the injection start/stop locations for necessary correction treatments, for example, by using geared pumping systems to dispense material with greater accuracy.

5 Discussion

The above results demonstrate the value of approximate, position-based fluid simulators in qualitatively predicting the behaviour of complex liquid sealants. In fact, the acquired cross-section views through simulation are otherwise inaccessible in the real world, which provides a clear benefit of simulation over real-world interactions: the robot can learn the underlying dynamics, rather than learning directly the action-outcome relationship as is common practice in machine learning, which would likely require less training data and be more transferable than traditional methods such as reinforcement learning.

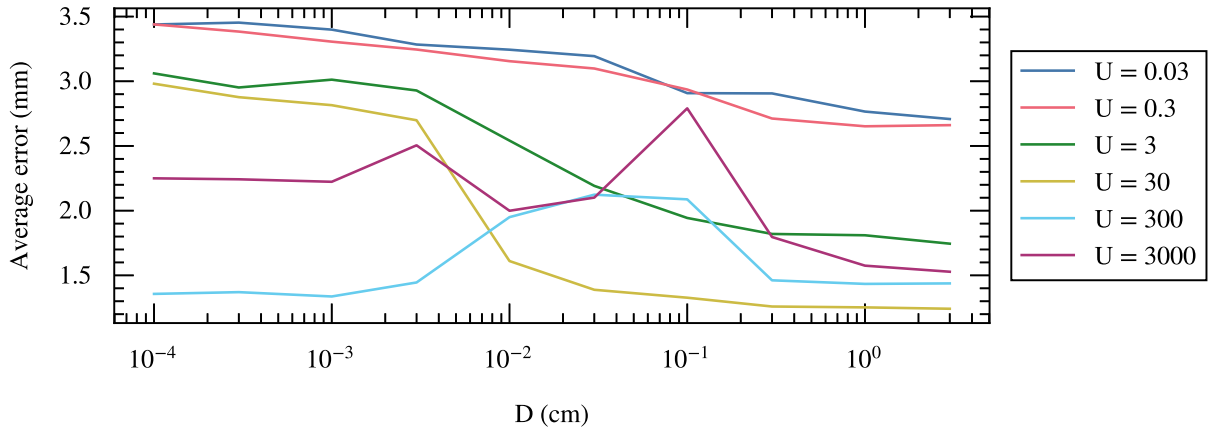
It should be noted that the only measured physical final fluid level in this work refers to the state when the sealant is completely settled in the crack and cured. So there should be a minor material flow and shrinkage behaviour (Lu et al., 2020), which can contribute to the simulation error. In addition, this study focused on results based on a single physical crack sealing trial, and attempts to emulate the observed dynamics. It is expected that more trials both with the same and different crack geometry samples would allow better quantification of the uncertainty in the PBF simulation. For the current PBF simulation, further discussions are provided below to establish a deeper understanding of its performance.

Influence of parameters U and D on simulation accuracy

In this work, the artificial viscosity parameters U and D are decisive for generating fluid behaviours, while they have demonstrated different influences on the simulation accuracy for different fluids. As is shown in Figure 10(a), D (wall drag) generally has a negative influence on the simulation error (despite a plateau stage when $D < 0.001$ cm): for the low-viscous fluids ($U < 1$), a limited negative influence is observed;

while such negative influence becomes more obvious for medium-viscous fluids ($1 < U < 100$); when the viscosity reaches a higher level ($U > 100$), the negative influence becomes limited again and also fluctuant. This is understandable and convincing: the wall drag can hardly work for water-like or solid-like fluids. U (fluid viscosity) is observed to have a fixed-mode, varying influence on the simulation error however strong or weak the interaction between the fluid and the wall is, as shown in Figure 10(b). This fixed-mode influence includes three stages: the plateau stage for $U < 1$, the negative correlation stage for $1 < U < 100$ and the positive correlation stage for $U > 100$.

(a)



(b)

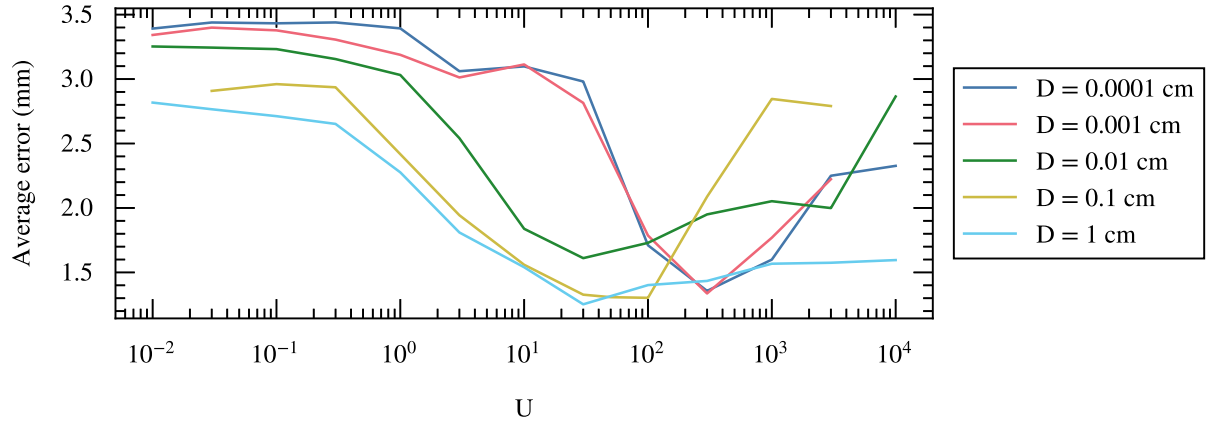


Figure 10: Simulation accuracy analysis: (a) The influence of D on the simulation accuracy; and (b) the influence of U on the simulation accuracy.

Approximation nature

The PBF method applies the approximation simulation principle which uses a few selected properties of the particle to depict the complex fluid dynamics. In this work, one important particle property, which is not considered in the current PBF algorithm, has been observed to be crucial for the success of simulation besides incompressibility and artificial viscosity (U and D). We call this property 'cohesion' which represents the convergence effect of a group of particles. Evidence is provided here: we add such cohesion property by purposely fixing the positions of the oldest particles released; observing the simulations, for most materials this appeared to have little impact as the oldest particles tend to be at the bottom of the fluid and unaffected

by the surface dynamics; when relaxing this motion constraint, allowing more particles to remain active, the fluid tends to settle to a flatter state. Therefore, only with the repelling step (incompressibility) and the slowing step (artificial viscosity), the fluid will settle given infinite time. By fixing the positions of older particles which show little motion, this acts as a very computationally cheap method of cohesion. However, being further removed from the physical dynamics this will directly lead to simulation failure, and a more accurate simulator could model this directly as an additional particle update step.

Computational power requirement

Under the computational power in this work, the number of simultaneously active particles is limited to 2,500 to allow practical simulation time, and moreover, there is a lower limit on the width of cracks that can be simulated. The simulation generally requires at least 4 particles along the width of the crack to allow effective flow of the particles. If we were to reduce the crack width, the particle diameter must be reduced to $d' = \frac{\text{crack width}}{4}$ to maintain the required number of particles along the width - that is, a scale factor of $\alpha = d'/d = \frac{\text{new crack width}}{\text{old crack width}}$. But if the length and depth of the crack are constant, the number of particles required to fill the crack increases by the factor of $\frac{1}{\alpha^2}$. Therefore, the number of particles scales very quickly. As Figure 11 shows, the required time to simulate each frame scales almost linearly with the number of fluid particles, leading to the difficulty/impossibility of simulating thin cracks quickly (e.g., reducing crack width by a factor of three increases simulation time by a factor of nine, so a 1 mm crack could be simulated in ∼54 minutes compared to the ∼6 minutes in this work). This can considerably slow the planning and control of physical robotic injection operations and so higher computational power is needed for sealing thin cracks.

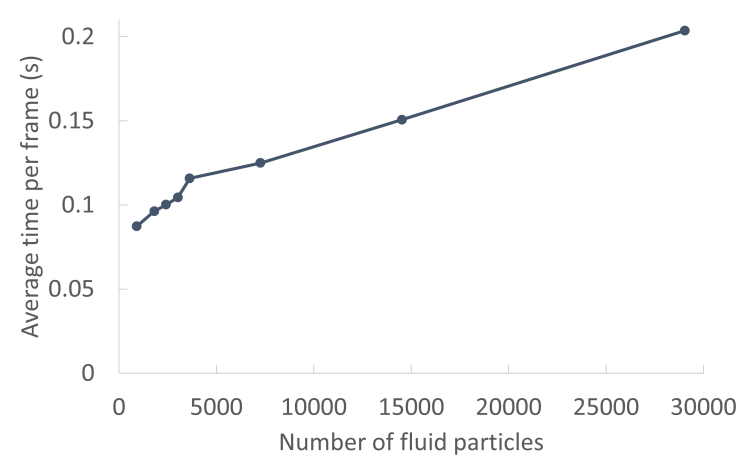


Figure 11: Relationship between the number of fluid particles and the required time to simulate one single frame.

Implications for future work

The search for the best-fit artificial viscosity parameters of the simulated fluids for their physical counterparts is the key to obtaining trustworthy simulation. This was subjectively conducted by the authors in this work and needs to be automated (e.g. through Bayesian optimisation or reinforcement learning), which may lead to a digital material calibration process. In addition, the time dependency of material properties (due to curing) also needs to be incorporated to better reflect the real-world dynamics.

With a perfect simulation of the material, the robotic injection parameters such as the nozzle trajectory and/or the nozzle flow rate can be optimised for achieving desired crack sealing quality with no overspill or underfill for the entire process. This allows a robot to establish an optimal action plan beforehand and then execute it open-loop. However, the main challenge of deploying this technique is to achieve such perfect simulation accuracy, due to the approximation nature of the PBF model and uncertain crack

geometries in practice. Rather than attempting to increase the accuracy of simulations, future work should focus on integrating feedback to update the simulation at regular intervals, as a model-predictive control (MPC) controller (Holkar and Waghmare, 2010; Todorov et al., 2012), to prevent the compounding effect of simulation inaccuracies.

6 Conclusion

This work demonstrates the implementation and evaluation of a fast, approximate fluid simulation, PBF, for its first use in a robotic crack injection sealing scenario by conducting a Real-to-Sim experiment. A physical cracked cementitious mortar sample was converted into a virtual model, deployed in the fluid simulation. Automated crack sealing using a SCARA robot with an injection syringe was performed in both the physical world and the virtual world. Building on an established PBF implementation, the simulator was expanded to model a range of fluids, and runs of the simulation were performed with 132 different sets of artificial viscosity properties to compare with the real sealant dynamics.

A satisfactory match has been observed between the simulated and physical crack sealant behaviour, with an average fluid level simulation error of 1.26 mm along the 80 mm crack length. It is found that the fluid viscosity parameter U has a fixed-mode three-stage (plateau-negative-positive) influence on the simulation error for all simulated fluids, while the wall drag parameter D generally has a negative influence on the simulation error especially for fluids with the viscosity range of $1 < U < 100$. These results thereby verify the feasibility of the PBF method and allow optimisation of the injection parameters (e.g. nozzle trajectory and flow rate) for planning robot actions to achieve quality crack sealing. However, it is also learned that the PBF method does apply the approximation principle and require a certain level of computational power to attain efficiency. Hence one should always balance the desired simulation accuracy and the effort (of particle definition) and time he/she put in the computation process. An MPC control mechanism can be potentially useful for compensating such simulation approximation in the practice of robotic crack sealing.

The ability to model hidden sub-surface flow of material is a key advantage of simulation over real-world experiments, as predictions can be made through physical reasoning rather than simple action-outcome relationships. Expanding such fast simulation/prediction with introducing learning to the framework for intelligent robots to allow autonomous identification, modelling and utilisation of physical dynamics to manipulate materials and interact with a dynamic environment. In addition to crack sealing, such an approach would be useful for any fluid manipulation tasks in construction, manufacturing, cooking and medicine such as 3D printing.

Acknowledgments

This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) [grant reference number EP/V056441/1] and the EPSRC Centre for Doctoral Training in Future Infrastructure and Built Environment: Resilience in a Changing World [grant reference number EP/S02302X/1]. The authors also want to thank Narges Khadem Hosseini for the lab support.

References

Al-Qadi, I. L., Fini, E. H., Elseifi, M. A., Masson, J.-F., and McGhee, K. M. (2006). Viscosity determination of hot-poured bituminous sealants. *Transportation research record*, 1958(1):74–81.

Alduán, I., Tena, A., and Otaduy, M. A. (2017). Dyverso: A versatile multi-phase position-based fluids solution for VFX. *Computer Graphics Forum*, 36(8):32–44.

Awuah, F. K. and Garcia-Hernández, A. (2022). Machine-filling of cracks in asphalt concrete. *Automation in Construction*, 141:104463.

Battaglia, P. W., Hamrick, J. B., and Tenenbaum, J. B. (2013). Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110(45):18327–18332.

British Board of Agrément (2010). Guidelines document for the assessment and certification of crack sealing systems for highways. Technical report, British Board of Agrément.

Chen, R., Zhou, C., and Cheng, L.-l. (2022). Computer-vision-guided semi-autonomous concrete crack repair for infrastructure maintenance using a robotic arm. *AI in Civil Engineering*, 1(1):9.

Choi, H., Crump, C., Duriez, C., Elmquist, A., Hager, G., Han, D., Hearl, F., Hodgins, J., Jain, A., Leve, F., et al. (2021). On the use of simulation in robotics: Opportunities, challenges, and suggestions for moving forward. *Proceedings of the National Academy of Sciences*, 118(1):e1907856118.

Fwa, T. F. (2005). *The handbook of highway engineering*. CRC press.

Guevara, T. L., Taylor, N. K., Gutmann, M., Ramamoorthy, S., and Subr, K. (2017). Adaptable pouring: Teaching robots not to spill using fast but approximate fluid simulation. In *1st Conference on Robot Learning 2017*, pages 77–86.

Guo, C., Yu, K., Gong, Y., and Yi, J. (2017). Optimal motion planning and control of a crack filling robot for civil infrastructure automation. In *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, pages 1463–1468. IEEE.

Haas, C. (1996). Evolution of an automated crack sealer: a study in construction technology development. *Automation in Construction*, 4:293–305.

Haas, C. T., Saidi, K., Cho, Y.-K., Fagerlund, W., Kim, H., Kim, Y.-S., et al. (1999). Implementation of an automated road maintenance machine (armm). Technical report, University of Texas at Austin. Center for Transportation Research.

Hawkins, J. (2008). PBD-fluid-in-Unity. [online] `https://github.com/Scrawk/PBD-Fluid-in-Unity` [Accessed: 2023-05-04].

Hensley, J. (2017). Tar snakes concern Indiana motorcyclists. [online] `https://hensleylegal.com/tar-snakes-concern-indiana-motorcyclists/` [Accessed: 2023-08-03].

Highways England (2017). Highways England highlights dangers faced by road workers. `https://www.gov.uk/government/news/highways-england-highlights-dangers-faced-by-road-workers`.

Highways Maintenance Efficiency Programme (2012). Prevention and a better cure - potholes review. Technical report, Highways Maintenance Efficiency Programme.

Holkar, K. and Waghmare, L. M. (2010). An overview of model predictive control. *International Journal of control and automation*, 3(4):47–63.

Holmes, J. (2010). Development of an automated pavement crack sealing system. Technical report, Georgia Institute of Technology - Department of Transportation, State of Georgia.

Hong, D., Velinsky, S. A., and Yamazaki, K. (1997). Tethered mobile robot for automating highway maintenance operations. *Robotics and Computer-Integrated Manufacturing*, 13:297–307.

Hosain, M. L. and Fdhila, R. B. (2015). Literature review of accelerated cfd simulation methods towards online application. *Energy Procedia*, 75:3307–3314.

Jackson, R. J., Patrick, P. S., and Miodownik, M. (2020). Functionally graded 3d printed asphalt composites. *Materials Letters: X*, 7:100047.

Jekel, C. F., Venter, G., Venter, M. P., Stander, N., and Haftka, R. T. (2019). Similarity measures for identifying material parameters from hysteresis loops using inverse analysis. *International Journal of Material Forming*.

Johnson, A. M. (2000). *Best practices handbook on asphalt pavement maintenance*. Minnesota Technology Transfer/LTAP Program, Center for Transportation Studies.

Lee, J. H., Yoo, H. S., Kim, Y. S., Lee, J. B., and Cho, M. Y. (2006). The development of a machine vision-assisted, teleoperated pavement crack sealer. *Automation in Construction*, 15:616–626.

Liu, J., Yang, X., Wang, X., and Yam, J. W. (2021). A laboratory prototype of automatic pavement crack sealing based on a modified 3d printer. *https://doi.org/10.1080/10298436.2021.1875225*, 23:2969–2980.

Lu, T., Li, Z., and van Breugel, K. (2020). Modelling of autogenous shrinkage of hardening cement paste. *Construction and Building Materials*, 264:120708.

Macklin, M. and Müller, M. (2013). Position based fluids. *ACM Transactions on Graphics (TOG)*, 32(4):1–12.

Müller, M., Heidelberger, B., Hennix, M., and Ratcliff, J. (2007). Position based dynamics. *Journal of Visual Communication and Image Representation*, 18(2):109–118.

Murphy, E. S. J. H. (2016). 'Black-tar snakes' causing serious motorcycle accidents nationwide. [online] `https://www.joneshacker.com/blog/2016/august/-black-tar-snakes-causing-serious-motorcycle-acc/` [Accessed: 2023-08-03].

Pan, Z., Park, C., and Manocha, D. (2016). Robot motion planning for pouring liquids. In *Proceedings of the international conference on automated planning and scheduling*, volume 26, pages 518–526.

Pioneer Industrial Systems (2022). RMV: Robotic crack sealer. `https://rmv.llc/`.

Schaefer, S., Palin, D., Hadjidemetriou, G. M., d'Avigneau, A. M., Al-Tabbaa, A., Iida, F., and George Thuruthel, T. (In Preparation). Robotic maintenance and repair of roads: A systematic review. *Automation in Construction*.

Seo, W.-J., Yoo, H.-S., and Kim, Y.-S. (2014). A comparative study on productivity analysis of automated pavement crack sealing machines. *KSCE Journal of Civil and Environmental Engineering Research*, 34(4):1289–1298.

Standards for Highways (2020). CM 231 - Pavement surface repairs. Technical report, Standards for Highways.

Takahashi, T. and Fujishiro, I. (2013). Accelerated viscous fluid simulation using position-based constraints. In *2013 International Conference on Computer-Aided Design and Computer Graphics*, pages 260–267. IEEE.

Todorov, E., Erez, T., and Tassa, Y. (2012). Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE.

Torbaghan, M. E., BEng, B. K., Abdellatif, M., Metje, N., BEng, J. L., MSci, R. J., Ing, C. D. F. R. E., Chapman, D. N., EngD, R. F., Miodownik, M., Richardson, R., of Robotics, F. P., Purnell, P., and of Materials, A. P. (2019). Robotic and autonomous systems for road asset management: a position paper. *Smart Infrastructure and Construction*.

Tsai, T. (2017). Position based dynamics. In *Encyclopedia of Computer Graphics and Games*, pages 1–5. Springer Cham.

Unity Technologies (2021). Unity user manual 2020.3 (LTS). `https://docs.unity3d.com/2020.3/Documentation/Manual/index.html`.

Velinsky, S. A. (1993). Fabrication and testing of an automated crack sealing machine. Technical report, University of California, Davis - SHRP.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Yang, X., Kahouadji, M., Lakhal, O., and Merzouki, R. (2022). Integrated design of an aerial soft-continuum manipulator for predictive maintenance. *Frontiers in Robotics and AI*, 9:980800.

Zhu, G., Fan, Z., Chen, W., You, Y., Huang, S., Liang, W., Fu, R., Xin, J., Chen, J., Deng, F., and Hou, Y. (2019). Design and implementation of a manipulator system for roadway crack sealing. *9th IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, CYBER 2019*, pages 1327–1331.