geoconvention
Calgary • Canada • June 20-22 2022

Well deformation: Simulation and reconstruction of multi-finger caliper observations

Mark Mlella[1] and Mirko van der Baan[1]
[1]Department of Physics, University of Alberta.

Summary

This study created a simulation that collected multi-finger caliper (MFC) data as the MFC tool moves through pipe meshes then analyzed different methods to reconstruct the pipe mesh geometries. The analysis was done with a goal of being able to process raw MFC data better. In order to do so, three dimensional (3D) modelling and simulation software were used to create a virtual MFC tool and pipe meshes for the analysis. Two ways through which pipe mesh data can be collected from the simulation were compared to test the efficacy of the simulation. Results from the comparison validate the use of the simulation for MFC data collection. Therefore, the simulation ease of use and ability to create a vast MFC dataset paves the way for its use in machine learning (ML) which requires a huge volume of data during training.

Introduction

The objective of this paper is to compare two ways through which pipe geometry data can be collected from a simulation. The comparison between the two different methods serves the purpose of validating MFC data collected from the simulation. Pipe mesh geometry obtained from Unity, a game engine, is referred to as the Unity derived data. While the pipe mesh geometry obtained after applying a geometric reconstruction work flow is referred to as the geometrically reconstructed data. Similarly, pipe geometry data obtained from Blender, a 3D modelling software, is referred to as Blender data mesh. The Unity derived data and the geometrically reconstructed data is compared by measuring the Euclidean distance between each data and the Blender data mesh.

The general work flow to achieve the comparison between the Unity derived and the geometrically reconstructed data is achieved using five steps (**Figure 1**). First, the pipe mesh geometries are modelled and generated in Blender. Then the Blender data is collected from the vertices of the Blender meshes. Second, the pipe meshes are imported into Unity. Then, the MFC tool is designed and instantiated in Unity then allowed to pass through the imported pipe mesh. The MFC tool is allowed to pass through the pipe mesh by free falling using gravity. This allowed an easy implementation as opposed to having a string pulling the tool through the pipe mesh as how it is done in reality. Third, the Unity derived data is collected as the MFC tool passes through the pipe mesh. Fourth, the raw MFC data collected as the tool passes through the pipe mesh is used to make the geometric reconstruction of the pipe mesh. Finally, an analysis between the Unity derived data and the geometrically reconstructed data is done to validate the data collected from the simulation. In the future, we will test if reconstruction of well geometries from MFC observations can be achieved using a machine learning approach.

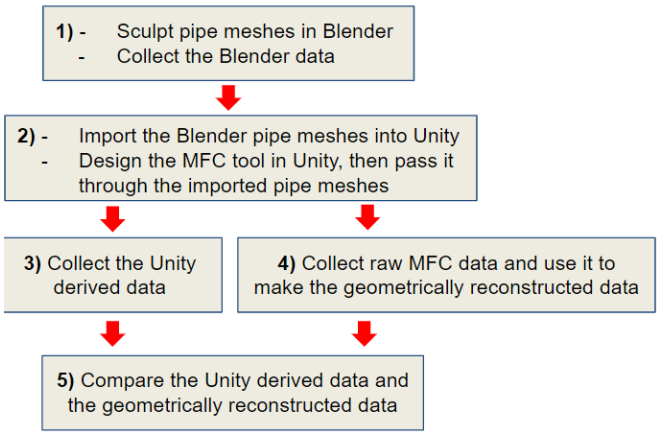

Figure 1: The general work flow used to achieve the comparison between Unity derived data and the geometrically reconstructed data.

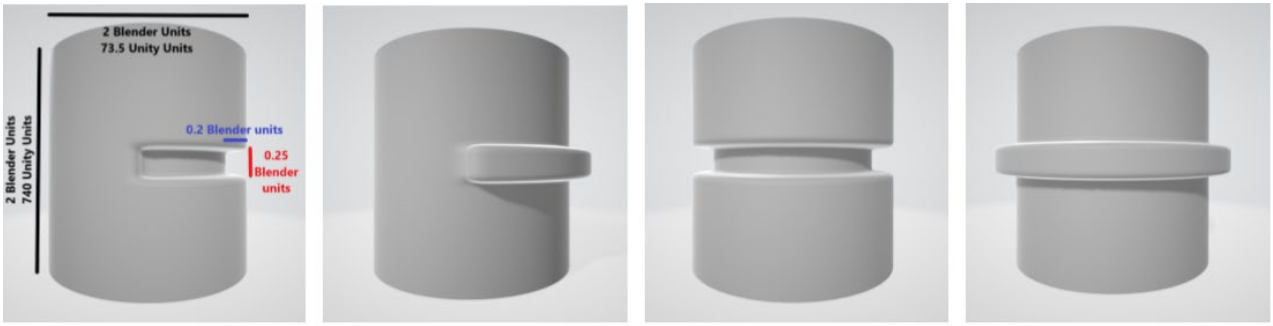

Figure 2: Pipe meshes with long wavelength used in the simulations, (a) asymmetric dent (b) asymmetric bulge (c) symmetric dent, and (d) symmetric bulge deformation.

Unity work flow

The Unity-derived and the raw MFC data is simultaneously collected as the MFC tool with 36 fingers moves through the pipe mesh (**Figure 2** (a)). The Unity-derived MFC data is made from the collision contact points between the MFC finger meshes and the pipe mesh through which the MFC tool is passed. On the other hand, the raw MFC data is also recorded as the simulation runs. The raw MFC data collected from Unity consists of: the center position of the tool, the angle each finger axis makes with the MFC tool axis, and the MFC tool Euler's orientation recorded by the gyroscope. Additionally, two more fixed parameters are used which are the MFC finger mesh length and the azimuth angle of each finger from the default finger. The raw MFC data is used as the input to the geometric reconstruction algorithm in order to yield pipe mesh geometry.

Unity derived MFC data

The Blender data mesh is used as the baseline for what is expected after processing MFC data collected from the simulation. The objective of collecting the Unity-derived MFC observations is for comparison with Blender data mesh. Sum of the Euclidean distance between each MFC finger

and the blender data mesh at every depth is used for the comparison. A good measure of the difference between the two mesh data is the Euclidean distance between them. The sum of the Euclidean distance between the Blender data mesh and the Unity derived data, denoted as ΔD, is defined as

$$\Delta D = \sum_{f=1}^{N} \sqrt{\Delta x_{f,d}^2 - \Delta z_{f,d}^2} \quad (1)$$

Where N is the number of MFC fingers, and $\Delta x_{f,d}$ and $\Delta z_{f,d}$ are the x-axis and the z-axis difference between the Blender data mesh and the Unity derived or geometrically reconstructed data points respectively at depth *d*.

Simulated MFC geometric reconstruction

The objective of developing a geometric reconstruction work flow is for using its results for comparison with the Unity derived data. Again, the sum of the Euclidean distance (ΔD), **Equation 1**, between the Blender data mesh and the geometrically reconstructed data is used to assess the quality of the pipe geometry reconstruction. In this study, a geometric reconstruction workflow has been developed to allow proper reconstruction of pipe meshes from the raw MFC data.

Results

Passing the MFC tool in deformed pipes in **Figure 2** result in the Unity-derived data in **Figure 3** (a - d). Similarly, sample reconstructed mesh obtained after applying a geometric work flow to the raw MFC data collected from simulation runs can be seen on **Figure 3** (e - h). Additionally, the Euclidean distance ΔD calculated using **Equation 1** between the Blender data mesh and the Unity derived and geometrically reconstructed data can be seen in **Figure 3** (i - l).

Discussion

The Unity derived data obtained from the contact points of MFC fingers and the pipe mesh has been observed to be jagged like in **Figure 3** (a - d). The jaggedness is a result of having multiple collision points between the MFC fingers and the pipe mesh. Also, it has been observed that some zones can have missing data as shown in **Figure 3** (a - d). When a finger fails to come into contact with the pipe mesh Unity returns the previous collision point as the current collision.

Geometrically reconstructed data (**Figure 3** (e - h)) can help solve the missing data problem observed in the Unity derived data. Also, the geometrically reconstructed data results in much more smooth and continuous data when compared to the Unity derived data. This is because the geometric reconstruction work flow assumes that the tip of the MFC finger makes contact with the pipe mesh at every depth step.

Similar to the Unity derived data, the geometrically reconstructed data yields ΔD that is generally below 10 Unity units for the undeformed zone and 13 Unity units for the deformed zone (**Figure 3** (i - l)). 13 Unity units is 18% of the pipe mesh diameter and it equates to an offset of 0.5% of the pipe mesh diameter for each of the 36 fingers used in the MFC tool. Therefore, the geometrically reconstructed data obtained provides a reliable pipe geometry reconstruction.

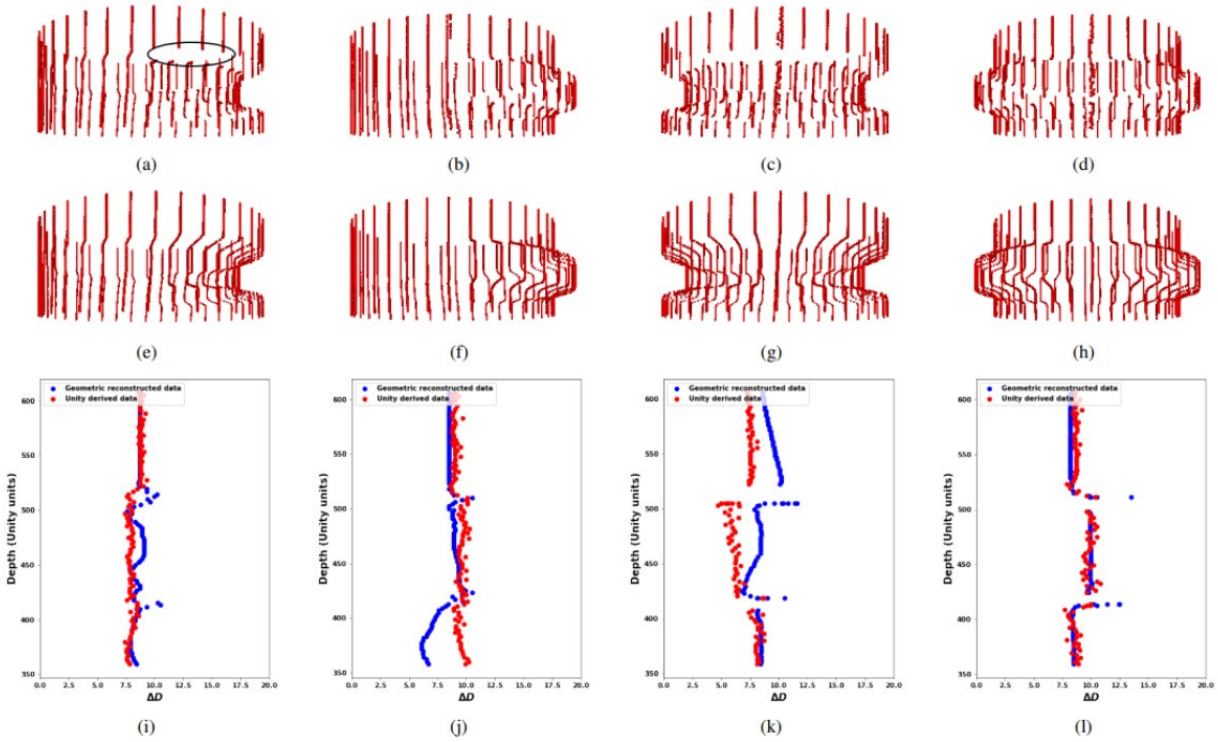

Figure 3: (a - d) Unity-derived and (e - h) geometrically reconstructed data collected after passing the MFC tool through pipe meshes (**Figure 2**) in the simulation. Notice the missing data points around the deformation zones, like in the area indicated by the black oval in (a). (i - l) the ΔD (i) asymmetric dent (j) asymmetric bulge (k) symmetric dent, and (l) symmetric bulge deformation calculated using **Equation 1**. The deformation is between 410 to 502 Unity units.

Conclusion

The MFC data collected using pipe meshes and MFC tool designed in 3D modelling software gives insight into pipe geometry reconstruction. The creation of pipe geometry data using the Unity derived data and the geometrically reconstructed data is validated because both data yield data within an acceptable range from the Blender data mesh which is the true mesh. Success in simulating MFC data make it possible to create large volumes of data which can be used to train ML models.

Acknowledgements

The author would like to thank the sponsors of the Microseismic Industry Consortium for financial support.