Update for Valeria

Jeremy Barisch-Rooney

October 4, 2019

Hi Valeria, here's a little update for you. It begins with the introduction taken from the thesis. Then a little overview of some work. Then a breakdown of the time and work involved in the thesis. I have included some questions for you, highlighted, and other parts may be highlighted. I am not sure if you have replied to Arpad's doodle for a meeting, but I believe you should have been sent an invite, let me know if you havn't received it.

This week's work has been going well, it is such a relief to have a quiet home to come home to. When I have the right environment I really enjoy my work, but when the environment isn't right work isn't so good. I worked until 4am on Sunday night, am happy to work as much as I can at the moment (not really knowing anyone in Delft is also probably a good thing too), told Arpi I will finish something tonight, so it could get late tonight as-well.

I am away for a weekend in October (to see a patron). And I will be away for a weekend for Christmas, will keep it short. I talked to the study advisor and she is happy to gave me an extension until the end of February, and she said if I want another extension that is fine too.

I am quite optimistic and motivated for this thesis, and would really appreciate your opinion on the stuff below, in particular your comments on the work and effort sections at the end of this document.

Also would you mind if I sent you a document outlining the sections and subsections (will change a lot I'm sure but will give me confidence about how I should be writing) of the thesis next week? Sometimes I am unsure if something is results/methods.

1 Thesis Introduction

The probability of a bridge to fail increases over time until it is no longer considered safe for use. Maintenance of a bridge is typically carried out when something goes wrong or according to a preventative maintenance schedule based on expert knowledge, neither approach making the best use of limited maintenance resources. Sensors can provide real-time information without the delay or cost of a manual maintenance check. The question of how to build an extensible decision support system (DSS) for bridge maintenance is answered in this thesis. What techniques do and don't work? And what are the costs and benefits of installing such a system on a bridge?

A **DSS for bridge maintenance** (BDSS) is a software system that provides the user of the system with information on the current status of a bridge. The provided information should enable the user of the system to make a more informed decision about when and/or where maintenance should be carried out. The provided information can include real-time sensor data and an analysis thereof. The primary aim of the analysis techniques applied to sensor data in this thesis is to classify data in two states, a normal or healthy state, and an abnormal or damaged states.

In the development of a BDSS it is **necessary to have sensor data** corresponding to both the damaged and undamaged states of the bridge in question. Due to the expensive nature of a bridge it is usually not allowed to impose a damaged state on the bridge. The use of finite element (FE) software allows us to create simulated sensor data via a finite element model (FEM) of a bridge on which different damage states can be imposed.

A FEM is a model of a structure in software. All models are wrong and thus the simulated sensor data generated by the FEM will **differ from sensor data collected from sensors installed in real life**. The simulated sensor data should be close enough to reality to provide confidence that the anaysis techniques in this thesis could also work with real sensor data, while acknowleding that some additional work would likely be needed to tune the anaysis techniques once real sensor data is available. After all, "in theory there is no difference between theory and practice, while in practice there is", TODO:REF.

A finite element simulation of a bridge can produce sensor data that is necessarily different from data collected from real installed sensors. It is necessary to **validate the simulated data against some ground truth** in order to have some confidence in the accuracy of the simulated data, and

it is also desirable to **test the developed analysis techniques on real data** such that we have some confidence in the techniques, for when a BDSS would eventually be. installed in real life.

A validated FEM of bridge 705 in Amsterdam was available at TNO for the FEM program Diana. This allows the programmatically generated FEM for OpenSees to be validated by comparing sensor data collected under a specified load from the OpenSees model of bridge 705 to sensor data collected from the validated Diana model under the same load.

TODO:Paragraph on classification

While we do not have bridge data availabe from a normal and abnormal state we do have data from viaducts for which there are two states available, high and low temperature. In this thesis the **analysis techniques are tested** on this data to provide, an albeit limited, test that the techniques can perform a classification between states.

Extensibility is a measure of the ability to extend a software system. According to TODO:REF "An important kind of reuse is extensibility, i.e., the extension of software without accessing existing code to edit or copy it." The research in this thesis is not just reproduceable but also extensible. The benefit of this research being extensible is that an interested party should be able to download the software and perform research on another bridge or investigate how a new classification technique performs on sensor data, without having to start from scratch. By creating an extensible system it is my hope to facilitate further research in this area and prevent duplication of effort.

For a software system to be extensible, the source code must be available to any user wishing to extend said software. The benefits of **open source software** are well known, in particular open source software allows *any individual with an interest* to develop or *extend* the software. Open source software can thus leverage the knowledge of the community and prevent duplication of efforts which can occur when software is developed behind closed doors. Open source software also provides transparency to anyone wishing to investigate the software and may produce more reliable software due to more people having eyes on it.

OpenSees (The Open System for Earthquake Engineering Simulation) is an open source finite element software framework that anyone with a Windows, macOS or Linux machine, and an internet connection, can download and install. Depending on open source finite element software enables users

to explore or extend the decision support software without requiring an expensive proprietary licence. In contrast to the Diana finite element software package, also used in this thesis, OpenSees does not require a licence for use and is additionaly available for macOS users.

This thesis first gives an overview of existing research into machine learning approaches for structural health monitoring (SHM), decision support systems and classification techniques. The methods section presents an in-depth description of how an extensible system is created for the collection of simulated sensor responses, how the inputs to this system should be structured, and what form the data-driven classification experiments will take. In the results section we take a look at the data generated by the data collection system, analyze the results of the classification experiments, and finally, present the costs and benefits of installing a decision support system for bridge maintenance.

2 Work

- I developed an extensible system for generating synthetic data from bridges under different bridge damage scenarios and different traffic conditions.
- The system has as parameter a bridge specification, bridge 705 is the parameter that is set right now for this thesis, bridge 705 details come from the Diana model. This parameter allows for not just bridge 705 to be analyzed, but any no pre-stress, no post-tension concrete slab bridge, which are most bridges I believe.
- The system has as parameter an interface to a finite element program, I call it a `FEMRunner`, in this thesis the only instance of this interface is targeting OpenSees, `OSRunner`. Work was done toward collection of data from the finite element program Diana for bridge 705 but that was scrapped as I am going for the open source FE program OpenSees to generate 3D results, if the interface to Diana had remained it would only be useful to collect results from an existing an Diana model (like bridge 705) under different traffic conditions.
- When we last talked only the 2D OpenSees model existed, the responses/bridge animations you saw were collected from that, the idea was that this "light" model would be used for quick collection of results (Diana is slooooow) and to compare how a "light" model compares to

a "good" model.

- Now I am using both 2D and 3D OpenSees models, the 3D model (hoping to finish it tonight) is being verified against models of bridge 705 in Diana and Axis. So the system supports 2D and 3D definitions of a bridge, for fast/slow data collection. And I can do a comparison of the quality/timing difference of the added dimension. 2D is about 1 second, 3D takes a few minutes. You can see the nodes of the generated FEM for bridge 705 in Figure 1.

- While developing the programmatic OpenSees FEM it is useful to validate the model built so far against a model in which there is some confidence, useful because it is not necessary to build the entire model before validation can occur. AxisVM allows for rapid building of FEMs via a GUI and inspection of the properties of the model and the responses from loading. The GUI in AxisVM allowed for the manual validation of a model of bridge 705. The AxisVM model was used to validate the OS 3D FEM during development, see contour plots of responses below.

  In Figure 2 and 3 you can see contour plots of loading from the AxisVM and OpenSees programmes, a bit more work needs to be done on the OpenSees model, before it is like the Diana one, namely the boundary conditions of the model (fixed nodes on supports and ends of the bridge deck) need to be set correctly, I have added the functionality to the system that generates the OpenSees model for setting how each support is fixed, and I am going to add the varying thickness of the piers and the deck tonight.

- The system allows for collection of "events" where an event is a time series of responses from one sensor under one simulation, at a certain frequency e.g. 250 Hz. It is really easy to request existing events or make new events under certain scenarios, or for example to request existing events from different sensors for the same simulation. Events can be collected under different `Trigger` s, this feature will likely go unused, a trigger says when to start or stop recording (perhaps due to some response threshold), currently only the `always_trigger`, which is always firing, is used. A stream of raw responses from a simulation are converted to a list of events, under some `Trigger`, using a `Recorder`, this could theoretically be hooked up to a stream of responses received via a HTTP server e.g. receiving live data. When we talked last I thought to have events be triggered by some condi-

tion/trigger as is done in one paper (Sydney Harbour Bridge (SHB)), but due to the high rate of traffic and that responses are possibly more localized in SHB, there will likely always be events triggered, hence always colelcting events. To avoid responses being split into two events at some time T where the split causes some useful feature for identification to be split in half, there is thus some overlap between events, see Figure 4.

- A bridge specification/input looks something like that below, I am using Haskell-like syntax here for writing down the bridge specification (Haskell has a nice syntax for describing data).

```
data Bridge = Bridge {
  name       :: String,
  length     :: Float,
  width      :: Float,
  supports   :: [Support],
  sections   :: [Section],
  lanes      :: [Lane],
  dimensions :: Dimension
  }

-- 2D or 2D model.
data Dimension = D2 | D3

-- A section describes a material e.g. thickness, young's modulus etc.
data Section = Section2D | Section3D

data Section3D = Section3D {
  density       :: Float,
  thickness     :: Float,
  youngsModulus :: Float,
  possonsRatio  :: Float,
  startX        :: Float -- Where the section start at x position.
}

data Section2D = Section2D {
  patches :: [Patch],
  layers  :: [Layer]
}
```

```
-- Not including patch and layer definitions here right now,
-- you can google OpenSees Patch or Layer if interested. The
-- Support2D is also missing.

-- Also missing in these definitions are the specification for
-- where sections begin/end (for the varying thickness of the
-- deck and pier).

data Support3D = Support3D {
  x           :: Float, -- x position of center of the support in meters.
  z           :: Float, -- z position of center of the support in meters.
  length      :: Float, -- length of the support in meters.
  height      :: Float, -- height of the support in meters.
  widthTop    :: float, -- width of the top of the support in meters.
  widthBottom :: float, -- width of the bottom of the support in meters.
  fixXTrans   :: Bool,  -- fix x translation of this support.
  fixYTrans   :: Bool,  -- fix y translation of this support.
  fixZTrans   :: Bool,  -- fix z translation of this support.
  fixXRotate  :: Bool,  -- fix x rotation of this support.
  fixYRotate  :: Bool,  -- fix y rotation of this support.
  fixZRotate  :: Bool   -- fix z rotation of this support.
}
```

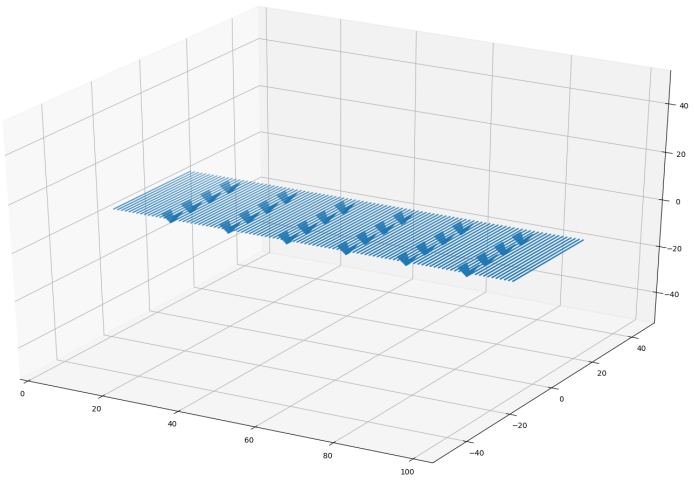

Figure 1: The nodes in the generated 3D FEM of bridge 705 for OpenSees.

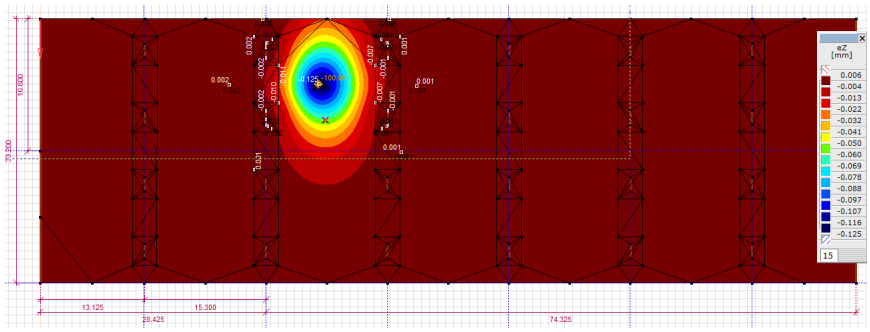

Figure 2: The responses of a model of bridge 705 in AxisVm to a load.

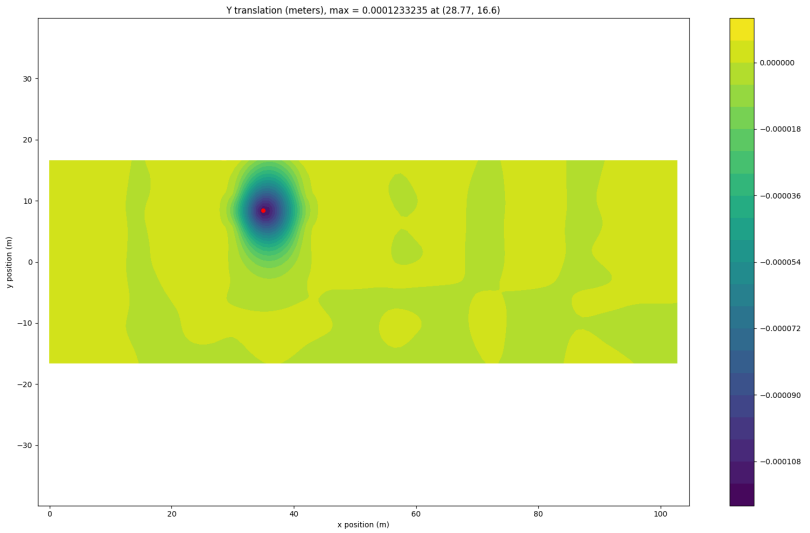

Figure 3: The responses of a generated model of bridge 705 in OpenSees to a load, the slightly darker green vertical "lines" are where the supports are.

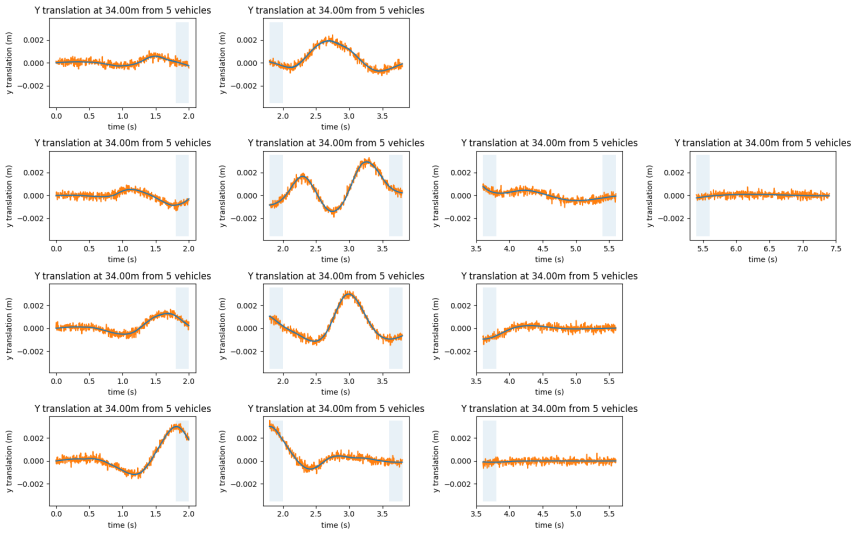

Figure 4: An example just to show that the functionality for collecting events is complete (don't ask me exactly what is going on though as it was a while ago and it will need to be revisited after next week). This is an early simplistic example from the 2D model and without much traffic.

3 Effort

Are you happy with the overview of the thesis components in the table below, or that it could be sufficient?

Is there anything you would particularly like to see?

Is there anything glaringly obvious missing? (I am particularly interested in an answer to this question) (I am aware it is really important to write down all the modeling assumptions, and that hasn't been mentioned yet e.g. how traffic scenarios and bridge 705 and damage scenarios are modeled).

Section	Name	Comment
FE model/system	2D model	Y translation, stress, strain. X translation need fixing.
FE model/system	3D model	Y, Z and X, translation, still needs varying thickness and pier displacement (a damage scenario and creating the infuence lines from responses. Being validated Axis and Diana.
Inputs/data	Vehicle data	Parameters for the system for bridge 705, taken from A16 data and NDW. A16 data is for heavy vehicles only, need to add some light vehicles.
Inputs/data	Bridge spec.	Bridge 705 is a parameter, dimensions taken from the Diana model.
Inputs/data	Noise	Ensure noise parameters (mean/stddev) come from real data from bridge 705 experiments.
Data model	Standard toolbox	There is a standard toolbox of classifier that should be tested, there are a number of different things to try here, can outline in person.
Data model	Calibration	If using MLP or similar calibrate/parameter tune for best performance.
Data model	Use information	On top of the standard toolbox make use of some structural bridge information.
Cost/benefit	-	This section largely motivates the work done, why it's useful etc, it is non-practical (little to no code) but a lot of writing/research.

I have left out some detail on the "Data model" section. There are a lot of relatively obvious things that can be tried here, different classifiers, try classify any deviation from the normal state, try to classify specific scenarios, try to detect location of damage, try combinations of models. What is the effect of increasing noise on classification (read: is it worth to buy more expensive sensors)? What is the effect on varying bridge parameters e.g. width, length, or number of supports? What is the effect of the density of sensors (these have an associated installation cost in real-life) on classification?

You can read the 4 sections in the table above as answering the questions:

- FE model/system: "I need a model to generate data to analyze and it should be extensible to target *many* bridges and allow for further research"
- Inputs/data: "I need to specify the data I want from the model, inputs being e.g. a bridge / a damage scenario / a traffic scenario"
- Data model: "I need to analyze the data generated by a FE model using a data-driven model"
- Cost/benefit: "how useful is all of this for a real-life bridge?"

Are you happy with the approximate breakdown of the thesis in the table below, or that it could be sufficient?

Section	Effort	Writing
(Extensible) FE model and data collection system (includes verified bridge 705 model)	40	30
Data-driven model	40	45
Inputs/data	5	5
Cost/benefit (applicability to different bridge types, cost and benefit of implementation in real what other techniques could work)	15	20

The FE model & data collection system and inputs/data have a lot done. The remaining ~5 months will be mostly on the data model (3months ?), cost/benefit (1month ?) and writing (1 month?). While I can keep an eye on the timeline a bit better, I really like how Arpi and I set weekly milestones (for two weeks in advance) recently, I think this is a nice pro-active approach. A combination of this pro-active approach and an eye on the broader targets is the way to go in my opinion.