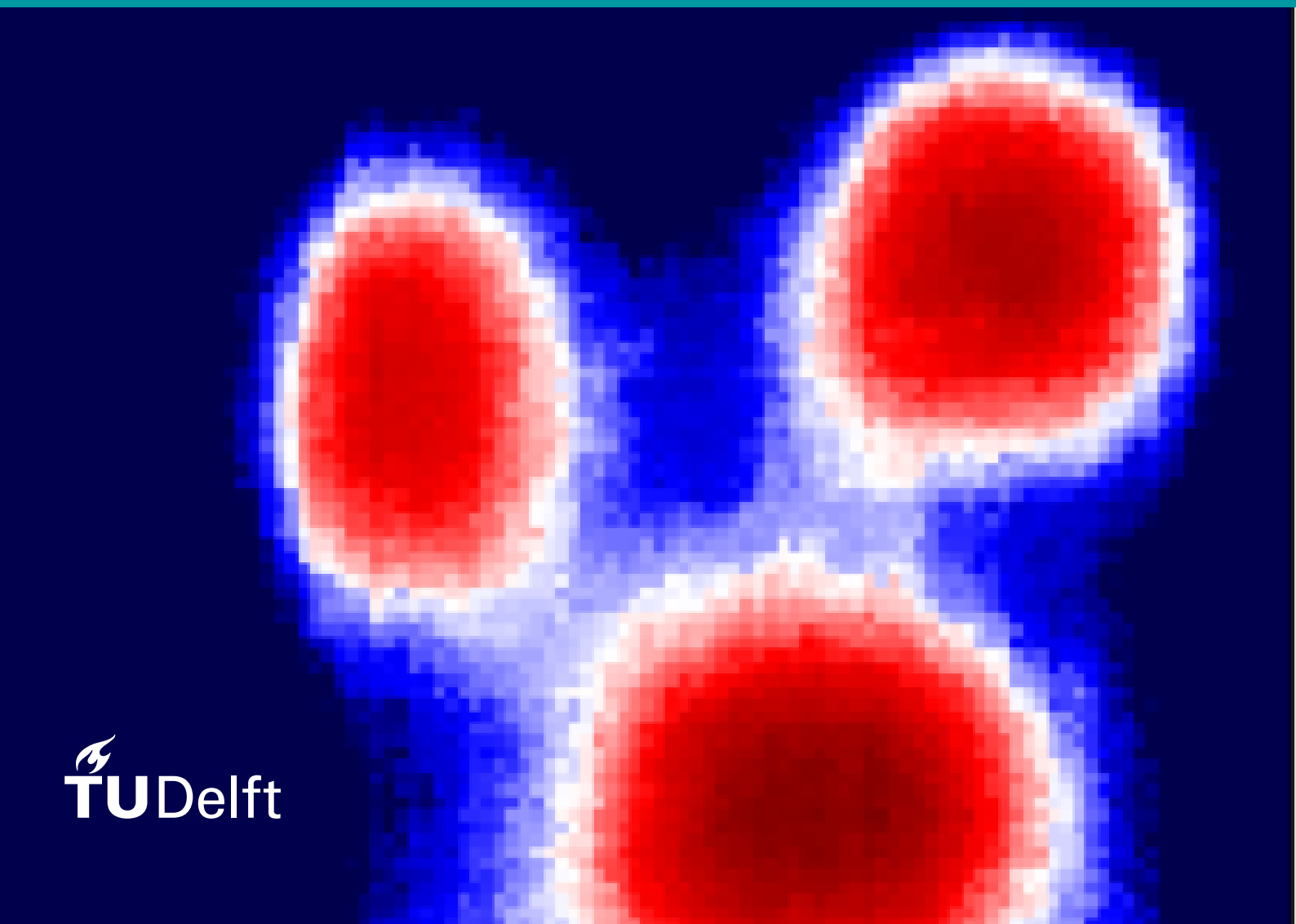
Potential of InfraRed Thermography for delamination detection in hybrid concrete structures.

A qualitative and quantitative study of delaminated areas between concrete and ultra-high performance concrete.

Maria Koetsier

Bachelor's Thesis
Civil Engineering

Picture of detected delamination from IRT data



TU Delft

Potential of InfraRed Thermography for delamination detection in hybrid concrete structures.

A qualitative and quantitative study of delaminated areas between concrete and ultra-high performance concrete.

by

Maria Koetsier

to obtain the degree of Bachelor of Science
at the Delft University of Technology,
to be defended publicly on Monday,June 21, 2021 at 15:00.

Student number: 4827635
Project duration: April 19th, 2021 – June 21th, 2021
Thesis committee: Dr. ir. M. Luković, TU Delft, Supervisor
Ir. E.R. van den Ham, TU Delft, Supervisor

TU Delft

Preface

To complete the Bachelor of Civil Engineering at the TU Delft, it is mandatory to complete one last part. This last part is writing the Bachelor's Thesis. The subject for this thesis it was important for me to really challenge myself with something new, therefore, the discussion I had with Dr. ir. M. Luković was ideal. She provided me with the option to perform a research into a new non-destructive testing method for detecting delaminated areas in a connection between a normal strength concrete beam and a layer of ultra-high-performance fibre reinforced concrete. She explained that this was something she was researching and that it was something totally new, which would mean that I would be doing new research. This was exactly what I was looking for to challenge myself.

In the process of writing this report it is assumed that the reader has basic knowledge about physics and about mathematics. This report is meant for people who are interested in the possibility of using active infrared thermography as a non-destructive testing method for detecting delaminations in hybrid concrete structures.

Readers who are interested in the theory behind the use of infrared thermography as a non-destructive testing method can find this in chapter 2. Readers who are interested the standardized approach and the analytical model which are created in this research can find its workings in chapter 6 and chapter 7 respectively. The readers who are interested in the experiments which are conducted can find this in chapter 3, the results from these experiments, computed by the analytical model, are shown in chapter 8.

I would like to thank my supervisors for their excellent guidance and their enthusiasm in the work I have done. I am grateful to Dr. ir. M. Luković for suggesting this research as a topic for my Bachelor's Thesis, for being always available for questions and for the useful feedback on my progress. I want to thank Ir. E.R. van den Ham for providing useful feedback on my progress, but mostly I want to thank him for sharing the knowledge about the building-physics side of this research, this was completely new to me and I am grateful that I was able to always ask questions.

Maria Koetsier
Delft, June 2021

Abstract

The concrete bridges which are constructed in the 1960s and 1970s, have been constructed without shear reinforcement. This means that these bridges do not comply with the current design regulations and might need an increase in shear capacity. A possible method to increase their shear capacity is to construct a layer of ultra-high-performance fibre reinforced concrete (UHPFRC) of a certain thickness on both sides of the normal strength concrete (NSC) beams supporting the bridge deck. An important issue to take into account when using this method is that the connection of the UHPFRC layer and the NSC beam needs to be of sufficient quality to transfer the shear stresses. This means that it is of interest to have a non-destructive testing (NDT) method for detecting possible delaminated areas within this interface.

This report investigates the possibility of using active infrared thermography (IRT) as an NDT method for detecting delaminated areas. The main goal of this report is to answer the following question: "Is it possible to detect and quantify delaminated areas, within the connection between a NSC beam and a layer of UHPFRC of a certain thickness in an objective way, with data gathered by an infrared camera in a lab environment?"

In this report a choice is made for how the delaminated areas will be detected and a standardized approach for testing samples is conducted. On top of that an analytical model is created to process the data gathered in the experiments, which were conducted following the created standardized approach. The data is gathered in the experiments with the use of an infrared camera. This camera measures the temperature of every pixel for every frame of a recording.

Literature study suggests three methods for detecting delaminated areas with active IRT:

- The thresh-hold value method; a threshold value is set for the temperature difference at the surface above a sound area and a delaminated area. When the temperature difference in an area is higher than this threshold value the area is delaminated.
- The signal to noise ratio (SNR) method; this method compares the temperature in an area to the temperature of a sound area and compares this to the STD of the temperature in a sound area, this is to take the environmental factors into account.
- The second derivative method; this method fits a 4^{th} order polynomial through the data gathered by the infrared camera. It does this for every row and column of the pixels in the picture. Each pixel has a temperature value attached to it, of the picture. Where there is an inflection point in the polynomial it is assumed that this indicates a boundary of the delaminated area.

The thresh-hold value method is very hard to make universal for different test settings, especially when the delaminated areas are unknown. This is why this method is not used in the current research.

Experiments have been conducted to determine which of the second derivative or the SNR method is more accurate. Both methods were compared and the following conclusions were drawn:

- The SNR method is more accurate about the surface area of the delaminated area than the second derivative method.
- The second derivative method overestimates the surface area of the delamination.
- The SNR method is easier to interpret than the second derivative method.

Due to these conclusion the choice is made to use the SNR method for detecting delaminated areas in the analytical model which is made.

The analytical model, which is made, detects delaminated areas of a specimen when a recording with a infrared camera is available. To be able to make the method of detecting the delaminated areas completely objective, a standardized approach for heating and cooling the specimen is constructed. Heating the specimen is done with a heat-flux of 1750 W/m^2 at the surface of the UHPFRC layer for 1500 seconds. The cooling process is then recorded with an infrared camera. With data gathered from this recording the analytical model takes out the optimal frame to investigate.

In the analytical model a distinction is made between two situations, one where the researcher knows the location of a sound area, and another situation where the researcher does not know the location of a sound area. For both situations the analytical model produced promising results. The simulated delaminated areas in the test samples of the experiments which were conducted in this research were visible in the results produced by the analytical model.

For the situation where there is no known sound area, the analytical model might underestimate the size or amount of delaminated area in the specimen. This was the case when testing a beam which was almost completely delaminated. The applicability and accuracy of the method for the situation where the researcher does not know the location of a sound area should thus be further researched. For both methods it is important to further investigate the minimal delaminated area which the analytical model is able to be detected.

Contents

List of Figures

List of Tables

Glossary

delamination thickness Thickness of the crack opening at the interface.. vii, 14, 16

heat capacity The amount of heat needed to increase the temperature of the material with one degree.. 4, 10

heat flux density The rate of heat transfer per unit area perpendicular to the direction of the heat transfer.. 4, 8

minimal delamination The smallest delaminated area which is of interest.. vi, 2, 3, 22, 35–39, 43, 58, 60

pulse-heating Heating with a heat source, which appears at one in full force, there is no gradual increase.. vii, 6, 7

thermal conductivity The rate at which heat passes through a material.. 4, 6, 10, 15, 56, 60

thermal diffusivity The ratio between the heat transfer and the energy storage of a material.. 4

Acronyms

FE Finite Element. 3, 4, 9, 31, 58

FNR false negative ratio. 45, 47, 49, 51, 54

FPR false positive ratio. 11, 45–47, 49, 51, 54

IRT infrared thermography. ii, iii, 1–4, 9, 11, 14, 15, 17, 35, 37, 38, 56, 58, 60

MRTC maximum running thermal contrast. 14

NDT non-destructive testing. ii, iii, 1–3, 9, 56, 60

NSC normal strength concrete. ii, iii, ix, 1, 2, 6, 9, 10, 17, 21, 22, 24, 31, 35, 56, 58

ROC receiver operating characteristic. 11

RTC running thermal contrast. vii, 10, 11, 14, 22, 32–37, 63

SNR signal to noise ratio. iii, v, vii, viii, 11, 12, 27–30, 32–34, 37–39, 43, 45, 46, 58, 59, 61

STD standard deviation. iii, 12, 27, 37–39, 43, 47, 48, 50, 52–55, 58

TNR true negative ratio. 45, 47, 49, 51, 54

TPR true positive ratio. 11, 45, 47, 49, 51, 54

UHPFRC ultra-high-performance fibre reinforced concrete. ii–iv, vii, ix, 1, 2, 6, 7, 9, 10, 19, 21–24, 31, 32, 35–37, 56, 58, 60, 63

1 Introduction

1.1. Project motivation

In the Netherlands, many concrete bridges have been built in the 1960s and 1970s. Those bridges have been constructed without shear reinforcement. Developments in traffic over the past years have been significant. The amount of trucks and the loading capacity of those trucks have increased and those trucks are governing for the design loads on bridges (Geesteranus, 2016) (Geesteranus2017). This increase in loading means that the concrete bridges from the 1960s and the 1970s might need an increase in shear capacity because they do not comply with current design regulations. A way to increase that shear capacity is to add a layer of ultra-high-performance fibre reinforced concrete (UHPFRC) of a certain thickness on both sides of the concrete beams supporting the bridge deck. Recent experiments show promising results in increasing the shear capacity of existing normal strength concrete (NSC) beams by gluing prefabricated UHPFRC to the side.
With this solution to increase the shear capacity of existing beams comes the problem of the quality of the UHPFRC layer's connection with the NSC beam. This connection's quality is an essential factor in the UHPFRC layer's performance to increase the shear capacity, because when a good bond is available between both layers the transfer of stresses is better. The prefabricated UHPFRC layers can be connected either by cast in situ concrete connection, dowels, gluing, etc. In this study, prefabricated elements are made and connected to existing concrete by an epoxy layer. An approach needs to be chosen to determine the quality of this connection. The best way to assess this quality is to be able to use non-destructive testing (NDT) methods. A solution to this question could be the use of infrared thermography (IRT). This solution is an NDT method that, according to recent studies (Kuhn et al., 2012) (Khan en Bartoli, 2015) (Huang et al., 2003), is promising in detecting delaminated areas in all kinds of materials. This method might also be applicable in this situation with concrete, IRT has already been used in research into an NDT method to detect defects and delaminations in concrete structures such as in research by Gu et al. (2020). However, this method is relatively new, and therefore there is no universal guideline on how to use it with different materials, geometry, exposure conditions, etc.. Therefore research has to be done on how to apply this NDT method for the situation stated above.

1.2. Problem statement

The detection of delaminated areas in the interface of two concrete layers is difficult to assess at a qualitative level, and it is crucial that during testing, the quality of the specimen does not change. Therefore it is essential to research NDT methods to assess the connection between the two concrete layers. NDT methods are characterized by the fact that the specimen's quality does not change and is still usable for its desired purpose after testing. IRT is a newer method that offers the availability to assess the connection between the two concrete layers. There have been multiple other kinds of research into IRT as an NDT method (Huh et al., 2018) (Kuhn et al., 2012). However, most of these researches had the goal of verifying IRT's capability to detect delaminated areas. Thus, many of those researched concentrated on the qualitative aspects of IRT, whereas the quantative aspects of IRT have not been thoroughly researched. (Omar et al., 2018) To determine the quality of the connection of the UHPFRC layer and the NSC beam it is important to be able to quantify delaminated areas. This means

that it is important to find a method to determine the delaminated areas' size and its boundaries. To understand how these aspects can be determined, research is needed into the theory, and multiple experiments are needed to verify and calibrate certain theoretical aspects. This study aims to create an analytical model to objectively predict delaminated areas' locations in the interface of a layer of UHPFRC and an NSC beam. The theory will be used to create the model, and multiple experiments will be used to calibrate this model and verify the accuracy of the model.

1.3. Scope

The research into the use of IRT as an NDT method for detecting delamination is very new, much is still unknown, and can become very extensive. This section will therefore discuss the scope of this research.

This research is conducted as a bachelor's thesis, therefore the duration of this research will be 8 weeks and will focus on one type of connection, a connection between NSC and a layer of UHPFRC where the layer of UHPFRC has the same thickness throughout the study. This research will investigate the following:

- Can delamination in the interface of a layer of UHPFRC and an NSC beam be detected with IRT?
- What is a good method/procedure to objectively determine delaminated areas?
- Is it possible to create a standardized approach for heating, cooling and measuring a specimen?
- Could an analytical model be created to objectively detect the delaminated areas?

In IRT different parameters are important to take into account, due to the short time period in which the research is conducted the effect of the following parameters are not taken into account:

- The thickness of the NSC, this is set at a 100mm.
- The thickness of the UHPFRC layer, this is set at 10mm.

Whereas the effect of the following parameters are taken into account:

- The thickness of the delaminated area.
- The area of the delaminated area.
- The heat-flux which is used.

1.4. Research questions

1.4.1. Main research question

"Is it possible to detect and quantify delaminated areas, within the connection between a NSC beam and a layer of UHPFRC of a certain thickness in an objective way, with data gathered by an infrared camera in a lab environment? In parallel to experimental study, a numerical study with commercial software COMSOL is performed to gain more insight into governing mechanisms and findings are compared to experimental results."

1.4.2. Subquestions

To answer the main questions the following subquestions will be answered.

- What is the theory behind the detectability of the delamination in 1D?
- What is a good heating time to create the highest thermal contrast for different thicknesses of delamination?
- What is the optimal cooling time to enhance visibility of the minimal delamination?
- What is a good and objective method to quantify the location and boundaries of the delaminated areas?

1.5. Research method

To gain understanding about the IRT method a literature review will be done into the 1D theory of IRT. With this 1D analysis the theory of the commercial Finite Element (FE) software COMSOL Multiphysics 5.6, hereafter referred to as COMSOL, can be understood and used. This software uses the same theory as the 1D analysis but has expanded this into a 3D model. With this software, experiments can be simulated, and data can be gathered, this data is needed to be able to create the model to determine the location of the delaminated areas.

The literature will be used to research existing methods to compute the optimal heating time to create the highest thermal contrast for different thicknesses of delaminated areas. Together with the 1D analysis and the COMSOL software, an optimal heating time will be computed and used in the experiments in this research.

The optimal cooling time of the specimen, to enhance visibility of the minimal delamination, will be determined by using data gathered from COMSOL simulations and by using data gathered from experiments that will be conducted. With this data the optimal cooling time can be derived.

For the location of the delaminated areas, an objective model will be made. This will be done with the knowledge gathered in literature study of previous research into IRT as an NDT method to detect delaminated areas as well as literature study of the 1D theory of the IRT method. The model will be calibrated to data gathered from recreated experiments in COMSOL. Once validated, the COMSOL model can also serve to perform parametric analyses and to investigate the role of governing parameters in more detail.

1.6. Reading guide

In chapter 2 a literature review is conducted. In chapter 3 the performance of the experiments is explained. Then in chapter 4 the method of detecting the delaminated area is chosen and explained. After this the use of the Finite Element software COMSOL Multiphysics 5.6 is explained and validated. Then in chapter 6 a standardized approach for performing the experiments is conducted. After that a model, which combines the standardized approach and the method for detecting the delaminated area is explained in chapter 7. In chapter 8 the results of this model are presented which are then discussed in chapter 9. Lastly, in chapter 10 the conclusion is given and in chapter 11 recommendations are made.

2

Literature review

In this chapter the important aspects of the to be created model will be researched, using literature. First, with the help from literature, the 1D theory of heat transfer will be explained. Then afterwards, literature research into previously done research into detecting delamination with IRT is conducted to learn what knowledge already exists about optimal heating times and the location and thickness of delaminated areas.

2.1. 1D Theory of heat transfer

In this section a general 1D analysis is briefly explained, this is to create an understanding of what the FE model COMSOL does. In this section the following source is mainly used: (TU Delft, 2000)

There are three ways for heat transfer to take place, namely:

- Conduction,
- Convection and
- Radiation

Heat transfer through conduction happens when heat is transferred from molecule to molecule. When heat is transferred through a flowing medium, then one speaks about convection. Radiation is the heat transfer in the form of electromagnetic waves.

2.1.1. Conduction

Conduction is the heat transfer through molecules. The heat flux density is shown in Equation 2.1 and the thermal diffusivity is given in Equation 2.2

$$q = -\lambda \frac{\delta T}{\delta x} \tag{2.1}$$

$$a = \frac{\lambda}{\rho c} \tag{2.2}$$

Where:
λ = thermal conductivity in W/mK
$\frac{\delta T}{\delta x}$ = temperature gradient in K/m
q = heat flux density in W/m^2
a = thermal diffusivity in m^2/s
ρ = density in kg/m^3
c = heat capacity in J/kgK
(ρc = volumetric heat capacity in J/m^3K)
Heat flows from higher temperature to lower temperature areas, this is why a negative temperature gradient results in a positive heat flux density. The thermal diffusivity gives the ratio between the heat transfer and the energy storage of a material and thus shows how good or bad the heat is diffused in a material.

In this analysis, one dimensional heat transfers are looked at and λ is considered to be constant in this study. This gives the differential equation given in Equation 2.3

$$\frac{\delta T}{\delta \tau} = a\frac{\delta^2 T}{\delta x^2} \tag{2.3}$$

When the situation is stationary, thus independent on time, the differential equation changes into:

$$a\frac{\delta^2 T}{\delta x^2} = 0 \tag{2.4}$$

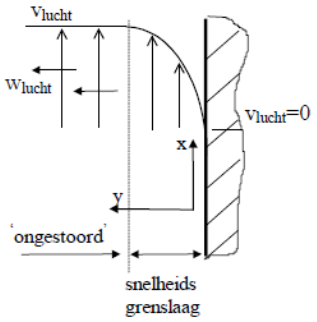
This means that:

$$\frac{\delta T}{\delta x} = constant \tag{2.5}$$

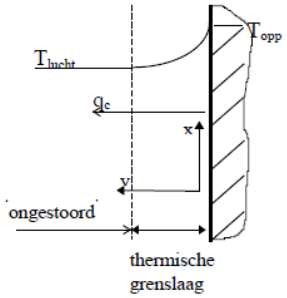
However, in the conducted experiments the situation is not stationary due to the heating source used to heat up the test samples. This means that the 1D analysis needs to be performed for non-stationary heat transfer.

2.1.2. Convection

When there is a difference in temperature between a surface and the air around it, heat transfer will take place. In Figure 2.1a and Figure 2.1b the situation is shown where there is a surface with a higher temperature than the ambient temperature. There will be a boundary layer where at the boundary the velocity of the air will be the same as the surrounding air and at the surface the velocity of the air will be zero. A thermal boundary layer will also arise, where the temperature at the surface will be the surface temperature, which will gradually decrease until the ambient temperature is reached. (TU Delft, 2000)



(a) Velocity gradient (TU Delft, 2000)



(b) Temperature gradient (TU Delft, 2000)

Figure 2.1: Temperature and velocity gradient due to convection

The heat transfer from a surface to a flowing medium due to convection can be defined as follows: (TU Delft, 2000)

$$q_c = \alpha_c(T_{opp} - T_{med}) \tag{2.6}$$

Where:
α_c = the convective heat transfer coefficient in W/m^2K
T_{opp} = the surface temperature in K
T_{med} = the temperature of the flowing medium in K

2.1.3. Radiation

Every surface emits a certain amount of thermal radiation, which is defined by its temperature. This thermal radiation is defined by the Stefan-Boltzmann law as follows:

$$E = \epsilon\sigma T^4 \tag{2.7}$$

Where:
E = the radiant emittance in W/m^2

σ = the Stefan-Boltzmann constant, equal to $5,67.10^{-8} W/m^2K^4$
ϵ = the emissivity coefficient of the surface
T = the temperature of the surface in Kelvin

The emissivity is a function of the wavelength λ, for a surface that has a maximal thermal radiance ϵ is 1, this is called a black body. 'Grey' surfaces have a value of ϵ between 0 and 1.

2.1.4. Air cavity

An air cavity gives an added thermal resistance to a structure due to its low value of thermal conductivity. The thermal conductivity of air is 0.025 [W/mK] whereas the thermal conductivity of NSC is 1.8 [W/mK]. One thing to keep in mind is that not only conductivity plays a role in the thermal resistance of an air cavity, also the radiance between the surfaces of an air cavity plays a role. The thermal resistance of an air cavity is defined as follows:

$$R_{cavity} = \frac{1}{\alpha_s + \alpha_g + \alpha_c} \tag{2.8}$$

Where:
α_s = The radiance heat transfer coefficient, which is 5.5 $[W/m^2K]$ at room temperature.
α_g = The conductive heat transfer coefficient, which is λ/d = 0.025/d $[W/m^2K]$, where d is the thickness of the air cavity.
α_c = The convective heat transfer coefficient, which is zero $[W/m^2K]$ for cavities smaller than 5mm.

An equivalent thermal conductivity can be calculated for different thicknesses of air cavities with Equation 2.9

$$\lambda_{eq} = 0.025 + 5.5d \tag{2.9}$$

Where d equals the thickness of the air cavity.

2.1.5. 1D theory applicable to this research

The experiment in the current research will consist of a multi-layered structure where the heating will consist of a pulse-heating with a constant heat-flux. For a one layered structure and a constant heat transfer of a constant temperature at the surface the following functions are defined (TU Delft, 2000):

$$T_W(x,\tau) = T_1 + (T_0 - T_1)erf(\beta) \tag{2.10}$$

Where:

$$erf(\beta) = \frac{2}{\sqrt{\pi}} \int_{\beta=0}^{\beta=\frac{x}{2\sqrt{a\tau}}} e^{-\beta^2}\, d\beta \tag{2.11}$$

Where:

$$\beta = \frac{x}{2\sqrt{\pi\tau}} \tag{2.12}$$

Where:
x = Depth in the layer [m]
τ = Time [s]
T_0 = Temperature before heating [°C]
T_1 = Heating temperature [°C]

This equation is very usefull when looking at one layered structures, however, they are not applicable to multi-layered structures. It is, however, possible to model the expected temperature over time at the backside of the UHPFRC panel when a pulse-heating excitation of a certain temperature is used. This is done in Figure 2.2.

pulse-heating indicates that a heat-source appears at a certain time in full force, it is not added gradually. The pulse-heating excitation of a certain temperature is different to the heating method used in this research, because in the research a pulse-heating excitation of a constant heat-flux is used. In

Equation 2.10 a pulse-heating excitation of a constant temperature is used. A constant heat-flux still creates an increasing temperature.

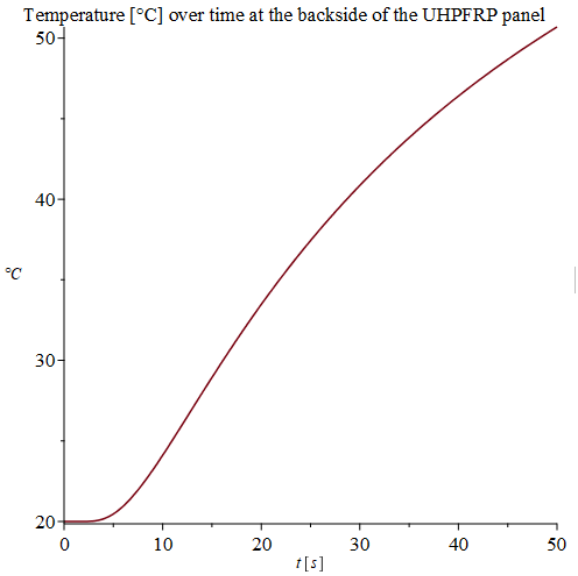


Figure 2.2: Modeled temperature in °C over time at backside of UHPFRC panel with a pulse-heating excitation of 100°C

For the multi-layered structure a matrix model needs to be made and the pulse-heating needs to be approached like a series of sines or cosines. Below the theory about the matrix model is explained:

One can model the total temperature with an average part and a fluctuating part as follows:

$$T_{tot}(x,t) = \overline{T} + \hat{T}e^{i\Phi}e^{i\omega t} \tag{2.13}$$

Where:
$T_{tot}(x,t)$ = Total temperature
$\overline{T}$ = Average temperature
$\hat{T}$ = Modulus of the non-time-dependent part
t = Time [s]
ω = Rotational frequency [rad/s]
Φ = phase shift [rad]

When the temperature has a sinusoidal gradient it is given that the fluctuating part of the temperature is:

$$\tilde{T} = Te^{i\omega t} \tag{2.14}$$

Substituting Equation 2.14 into Equation 2.3 gives:

$$i\omega T(x)e^{i\omega t} = ae^{i\omega t}\frac{d^2T(x)}{dx^2} \tag{2.15}$$

Then the general solution to Equation 2.15 is:

$$T(x) = Ae^{x\sqrt{\frac{i\omega}{a}}} + Be^{-x\sqrt{\frac{i\omega}{a}}} \tag{2.16}$$

Take the homogeneous wall presented in Figure 2.3:

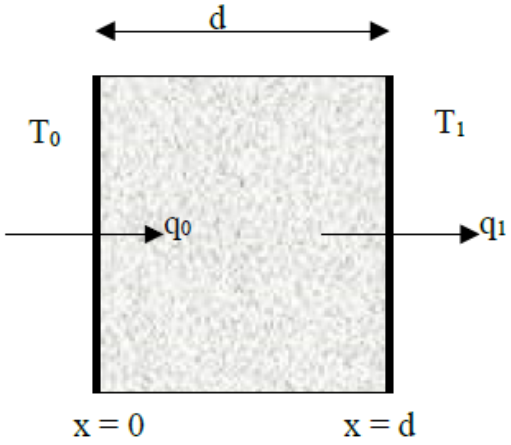


Figure 2.3: Homogeneous wall (TU Delft, 2000)

T_0 = non-time-dependent part of the temperature fluctuation at x = 0
T_1 = non-time-dependent part of the temperature fluctuation at x = d
q_0 = non-time-dependent part of the heat flux density at x = 0
q_1 = non-time-dependent part of the heat flux density at x = d

In (TU Delft, 2000) from Equation 2.16 the following relationship between T_0, T_1, q_0 and q_1 in matrix-form is deducted:

$$\begin{pmatrix} T_1 \\ q_1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} T_0 \\ q_0 \end{pmatrix}$$

With:
$a_{11} = a_{22} = \cosh(1+i)kd$
$a_{12} = -\frac{\sinh(1+i)kd}{\lambda(1+i)k}$
$a_{21} = -\lambda(1+i)k \sinh(1+i)kd$
where:
$k = \sqrt{\frac{\omega}{2a}}$

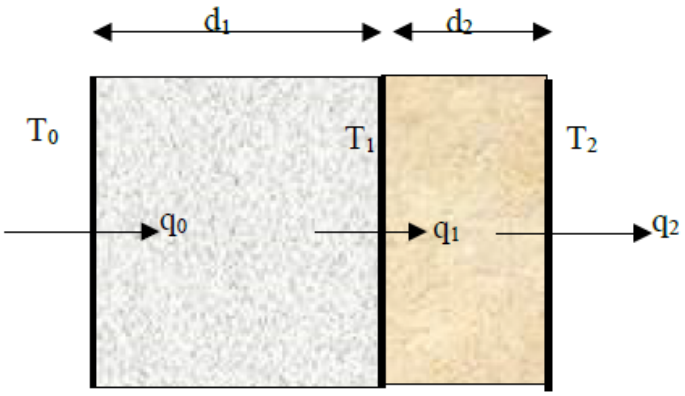


Figure 2.4: Wall with multiple layers (TU Delft, 2000)

When a wall consisting of multiple layers, as shown in Figure 2.4, is examined, the following equa-

tions can be found:

$$\begin{pmatrix} T_1 \\ q_1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} T_0 \\ q_0 \end{pmatrix} and \begin{pmatrix} T_2 \\ q_2 \end{pmatrix} = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} \begin{pmatrix} T_1 \\ q_1 \end{pmatrix} \tag{2.17}$$

Substitution gives:

$$\begin{pmatrix} T_2 \\ q_2 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} \begin{pmatrix} T_0 \\ q_0 \end{pmatrix} = \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix} \begin{pmatrix} T_0 \\ q_0 \end{pmatrix} \tag{2.18}$$

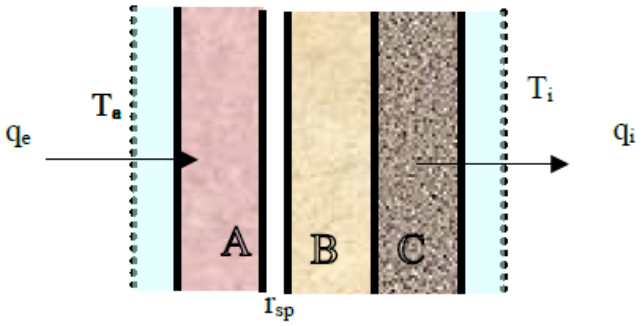


Figure 2.5: Wall with unventilated cavity (TU Delft, 2000)

Now looking at a wall structure with an unventilated cavity as seen in Figure 2.5, the following expression is given in (TU Delft,2000):

$$\begin{pmatrix} T_i \\ q_i \end{pmatrix} = \begin{pmatrix} 1 & -\frac{1}{\alpha_i} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} \begin{pmatrix} 1 & -r_{sp} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} 1 & -\frac{1}{\alpha_e} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} T_e \\ q_e \end{pmatrix} \tag{2.19}$$

Where $\alpha_i = \frac{-d}{\lambda}$ and r_{sp} is the heat resistance of the unventilated cavity.

This one dimensional analysis is a short explanation of what FE models use in a three dimensional analysis. In the following part of this report the FE software COMSOL will be used instead of a one dimensional analysis. FE simulation is more accurate, because it is a three dimensional model that does take into account the lateral heat flow as well, in contrast to a one dimensional model. It is however important to understand the basic one dimensional analysis as shown above.

2.2. The use of Finite Element Modelling

In this research FE modelling will be used to determine certain important aspects of the experiment configuration.
The choice has been made to use COMSOL Multiphysics 5.6. In some similarly aimed studies the commercial FE software COMSOL has also been used to simulate the distribution of heat throughout a test specimen. For example in research by Junyan et al. (2013) where IRT was researched as an NDT method to detect subsurface damage in carbon-carbon composites. The research by Junyan et al. (2013) used pulse and modulation heat excitation conditions.
Also in research by Hiasa et al. (2016) the FE application COMSOL was used to predict the temperature above delaminated and sound areas, wheras the exact temperature was hard to simulate, the difference in temperature was good to estimate. In that study the use of COMSOL was mostly used to create a specific scale in which to display the IRT image so that the delaminated areas are no longer determined based on subjective criteria but rather objective criteria.

2.3. Physical properties of NSC and UHPFRC

Several physical properties of NSC and UHPFRC are needed in this research. The needed properties come from the one dimensional theory explained in the previous section. The physical properties of

NSC taken from the standard concrete material of COMSOL. The volumetric heat capacity ρc and the thermal conductivity λ of UHPFRC are are taken from research by Nagy et al. (2015), these physical properties are determined for seven samples. For this study the average of the values found by Nagy et al. (2015) are taken. For the emissivity ϵ of UHPFRC the same value is taken as for NSC, because the emissivity is dependent on the colour of the material, which is rougly the same for both materials. In Table 2.1 the physical properties of both materials are given.

	NSC	UHPFRC
Volumetric heat capacity $\rho c[J/m^3K]$	2,024.10^6	2,14.10^6
thermal conductivity $\lambda[W/mK]$	1,8	2,82
Emissivity $\epsilon[-]$	0,93	0,93

Table 2.1: Physical properties of NSC and UHPFRC

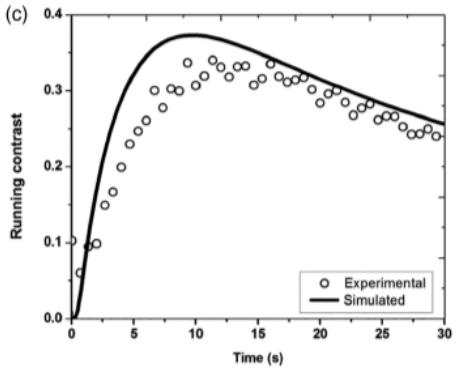
2.4. Optimal heating time

To detect the delaminated areas the samples will be heated externally. Due to heating the specimen, the temperature increase of the outer layer of the sample will increase more above a delaminated area than above a sound area, this is due to the thermal resistance of the layer of air inside the delamination. It is found that the optimal heating time is to be determined such that the difference in temperature reaches a maximum (Sinha et al., 2015). This difference in temperature, the contrast, is what makes the delamination detectable, therefore it is important to increase the relative contrast, hereafter called the running thermal contrast (RTC), as much as possible (Lai et al., 2010).
The RTC is defined as follows:

$$\Delta T_r = \frac{(T_{def} - T_{non-def})}{T_{non-def}} \tag{2.20}$$

Where:
T_{def} = The temperature above a delaminated area.
$T_{non-def}$ = The temperature above a sound area.

In research by Sinha et al. (2015) experiments were conducted with delaminated areas at a certain depth in Photo-voltaic panels. In these experiments a halogen lamp was used as to create a step heating heat source, where it was placed, such that it created a heat flux of 1000 $[W/m^2]$ at the surface of the Photo-voltaic panel. In that research it was chosen to determine the optimal heating time through a simulation and compare this to the experiment data gathered. As shown in Figure 2.6a , the time at which the RTC reaches a maximum according to the simulation was quite accurate compared to the experiment data. The test setup of the experiments performed by Sinha et al. (2015) is shown in Figure 2.6b



(a) Simulated and experimental plots of the RTC by using step heating with a halogen lamp to create a heat-flux of 1000 $[W/m^2]$ (Sinha et al., 2015)

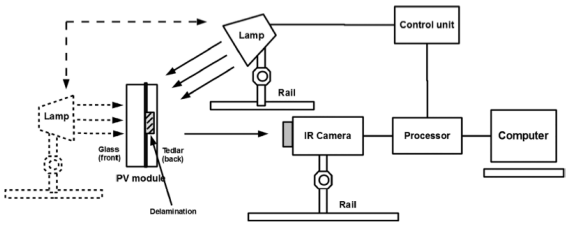


Figure 2. Schematic of experimental set-up for active thermography.

(b) Setup of the experiments performed by Sinha et al. (2015)

Figure 2.6: Results and setup of experiments by Sinha et al. (2015)

2.5. Location of delaminated areas

When the optimal heating time is determined it is possible to conduct the experiments. One of the goals of these experiments is detecting the location of the delaminated areas. Most of the research into IRT has focused on determining the location of the delaminated areas. There are three methods which were mostly used in previous researches. These method are the threshold value method, the SNR method and a method where the second derivative is taken to detect inflection points. In this section these methods are further elaborated.

2.5.1. Threshold value

The threshold value method is a method where a threshold value for the thermal contrast is chosen. Whenever a pixel has a value higher than this thermal contrast, this indicates that at the place of this pixel a delamination is present (Sinha et al., 2015). To determine the reliability of the threshold value a receiver operating characteristic (ROC) analysis can be performed, because it is able to provide a universal measure of reliability. The reliability of diagnostic systems have been widely determined by the ROC analysis method (Metz, 2006). In the ROC analysis method a graph is constructed on which the true positive ratio (TPR) and the false positive ratio (FPR) are presented for different threshold values. On the x-axis the FPR is presented and on the y-axis the TPR is presented. In Figure 2.7 an example of an ROC graph, all the black dots represent a different threshold value. It is possible to choose threshold values to preferences in TPR and FPR. For a balance in the TPR and the FPR the threshold value closest to the point (0,1) needs to be chosen. This point should give relatively high TPR and low FPR. A big downside to this method, is that the actual delaminations need to be known, or the threshold temperature needs to be conducted from experiments with the same circumstances, because otherwise it is impossible to determine the FPR and TPR rates at the certain threshold values. Due to this big downside it is hard to determine the right threshold values in different cases where the delaminated areas are unknown, this is why this method will not be used in the current research.

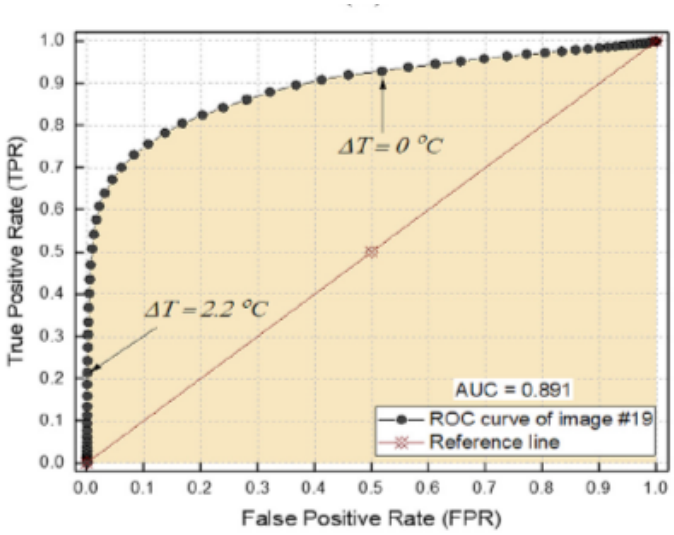


Figure 2.7: ROC graph (Metz, 2006)

2.5.2. Signal to noise ratio

Another way to determine the boundaries of the delaminated areas is to use the signal to noise ratio (SNR) method. This method takes into account the noise in the infrared data. This noise, created due to environmental factors, is also present when looking at a sound area. With the SNR method it is possible to take this phenomenon into account. The signal to noise ratio is defined in Equation 2.21 (Wang et al., 2020).

$$SNR = 20log_{10}\frac{|S_d - S_a|}{\sigma_{S_a}} \tag{2.21}$$

Where:
S_d = the average temperature above the delaminated area
S_a = the average temperature above the sound areas
σ_{S_a} = the standard deviation of the sound areas

When the SNR has a value above 0, it means that in this area the temperature difference between the area and a sound area is higher than the standard deviation (STD) of the sound area, which is why this area is then classified as delaminated (Wang et al., 2020).

2.5.3. Second derivative method

A third method to detect delaminated areas, that is mentioned in literature, is to use the second derivative of the temperature graph. It is possible to detect inflection points when searching those places where the second derivative of the temperature graph is zero. To be able to do this with data gathered from an infrared camera, it is needed to first smooth the graph produced by the camera, to remove the inconsistencies in the measurements. In Figure 2.8 an example of this method is shown and in Figure 2.9 the test setup of the results shown in Figure 2.8 is shown.

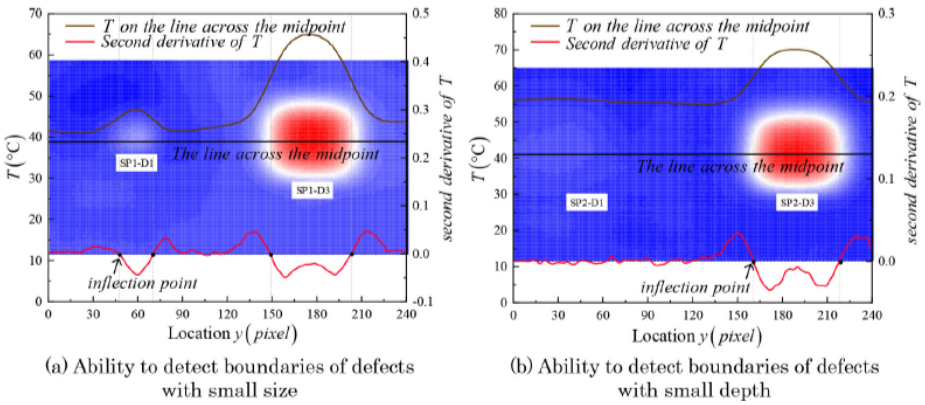


Figure 2.8: Smoothed temperature graph and its second derivative with defected areas (Gu et al., 2020)

Fig. 4. Setup of the experiment.

Figure 2.9: Test setup by of experiments by Gu et al. (2020)

However, as can be seen, the inflection points do not only occur above delaminated or defected areas. This means that there would need to be an additional check to determine whether there really is a delaminated area or if this is caused by noise in the measurements.

This method was also used in research by Lai et al. (2010), however in this research both flaws and delaminations were tested, where in this case flaws were air cavities with sharp edges and delaminations were air cavities with curved edges, this is shown in Figure 2.10. It was researched whether the boundaries of the flaws and delaminations of this method, using the second derivative, were representative for the real boundaries of the flaws and delaminations. It was found that for the boundaries of the flaws the accuracy was 88%, in contrast to the boundaries of the delaminations, where it was found that the results from the analysis were not consistent with the real boundaries of the delaminations.

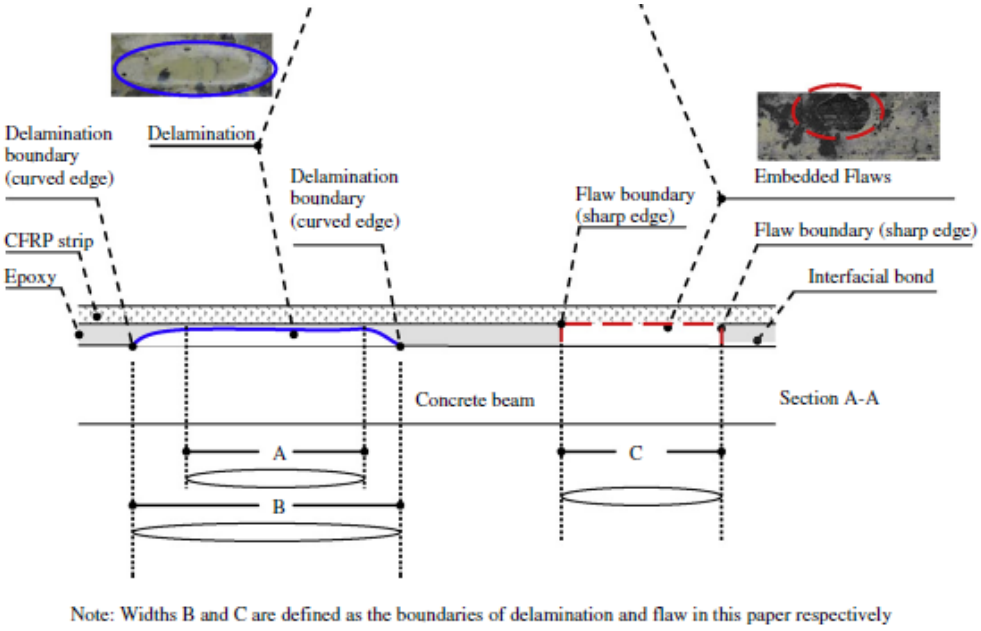


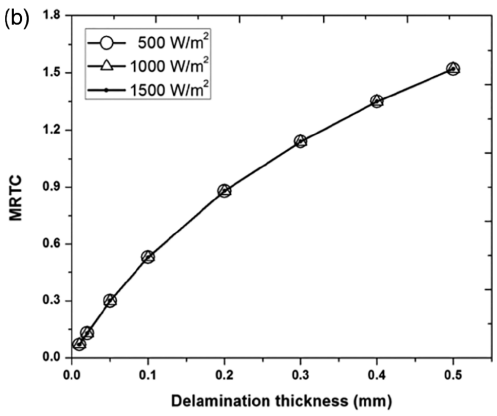
Figure 2.10: Embedded flaws and delaminations in research by Lai et al. (2010)

2.6. Thickness of delaminated areas

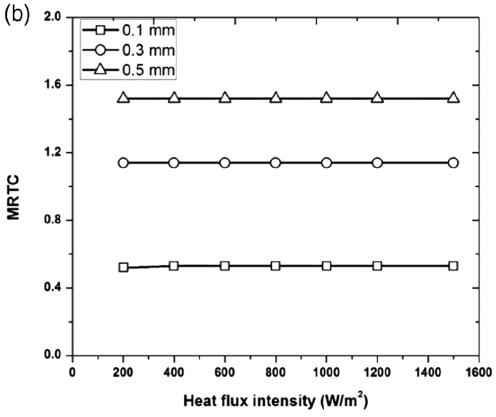
Previous researches into the use of IRT for delamination detection have mostly focused on detecting delaminated areas, not on measuring the thickness of delaminated areas. Nevertheless, there have been some studies where the thickness of delaminated areas has been researched.
In the current research the delamination thickness is important To investigate the effect of the thickness on the detectability of the delaminated area.

In (Sinha et al., 2015) research has been done into determining the delamination thickness in Photovoltaic panels, because the depth of the delamination was known this was advantageous. It was found that the maximum running thermal contrast (MRTC), which is the maximum value of the RTC of a delaminated area, per delamination thickness was constant with different heat flux intensities, as shown in Figure 2.11b.

When using the MRTC this then means that the heat flux intensity has no effect on the estimation of the delamination thickness. Because the depth of the delamination was known, and the MRTC is used, it was possible to plot the MRTC over the delamination thickness. After that a curve was fitted through these data points, this curve can then be used to determine the delamination thickness belonging to a certain MRTC value. This constructed graph is presented in Figure 2.11a



(a) MRTC over delamination thickness, data points and fitted curve (Sinha et al., 2015)



(b) MRTC for different thicknesses and heat flux intensities in PV panel (Sinha et al., 2015)

In research by Guillaumat et al. (2004), another method to determine the thickness of the delaminated area was used. It was stated that the thickness of the air layer can be approximated as follows:

$$\frac{e}{\lambda_{eq}} = \frac{e}{\lambda} + \frac{e_{air}}{\lambda_{air}} \tag{2.22}$$

Where:
e = the global thickness (assumed to remain unchanged) [m]
λ_{eq} = the equivalent thermal conductivity [W/mK]
λ = the undamaged sample thermal conductivity [W/mK]
λ_{air} = the thermal conductivity of air [W/mK]
e_{air} = the thickness of the air layer [m]

Note that to determine the thickness of the air layer, first the equivalent thermal conductivity needs to be determined through experiments. It is also stated that this tool is not accurate but can be used for comparisons or different interpretations such as delamination density (Guillaumat et al., 2004). This can be undestood with the theory explained in the previous subsections because the radiative heat transfer is not taken into account in Equation 2.22. Also it would probably be impossible to derive thicknesses smaller than a millimeter with this method, because the overall thermal conductivity of the structure would increase only slightly.

Even though these previous mentioned studies did find a way to estimate the thickness of delamination with IRT, in another study by Hiasa et al. (2016) it was found that the thickness of the delamination within concrete structures had no significant effect on the temperature difference between sound areas and delaminated areas, this can be seen in Figure 2.13. In the experiments Hiasa et al. (2016) performed, the sun was used as heating source and concrete blocks have been made with artificial delaminations. The test specimens were ”manufactured concrete blocks which had artificial delaminations at different depths from the surface, 1.27 cm (0.5 in.), 2.54 cm (1 in.), 5.08 cm (2 in.) and 7.62 cm (3 in.). The dimensions of each delamination were 10.2 cm (4 in.) x 10.2 cm x approximately 0.3 cm (1/8 in.). All four concrete blocks had the same dimensions as 91.4 cm (3 ft.) x 91.4 cm x 20.3 cm (8 in.), and the thickness was designed to simulate a typical bridge deck in the USA.” (Hiasa et al., 2016). The test setup for this experiment can be seen in Figure 2.12

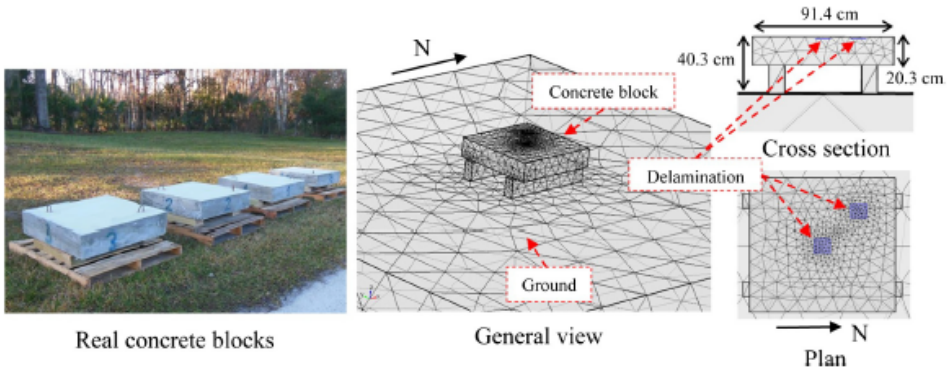


Figure 2.12: Test setup by of experiments by Hiasa et al. (2016)

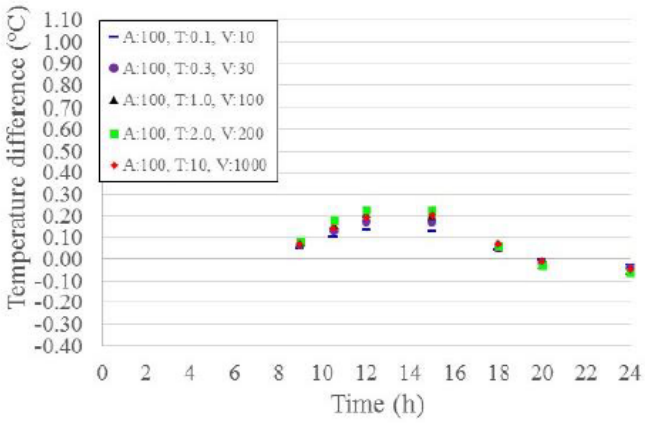


Figure 2.13: Effect of delamination thickness on temperature difference in research by Hiasa et al. (2016)

In this figure, A means area [cm^2] of the delamination, T means thickness [cm] of the delamination and V means volume [cm^3] of the delamination

3

Experiments

For this research multiple experiments have been conducted with different test samples. In this chapter the conducted experiments are discussed, the method of the experiment as well as the type of test sample. The experiments that have been conducted are in order from simple to more advanced. The following experiments have been conducted in the presented order:

- Proof of concept experiments; small experiments with simulated delaminations to find the influence of different factors.
- Main experiments; experiments with simulated delaminations of different thicknesses on different areas in the sample.
- Blind experiments; experiments with simulated delaminations where the author did not know the size or location of the delaminations.
- Real experiment; experiment on a beam where the delamination is partly unknown.

These experiments will be explained below, in the above given order.

3.1. Proof of concept experiments

The first experiments conducted had the goal to see if the concept of detecting delaminated areas with IRT is possible. Multiple small tests have been conducted with different configurations.

3.1.1. Test samples

For the test samples two available NSC blocks of 10x10x10 [cm] were used. On top of those samples available lamella of 10mm thickness were glued with an epoxy glue. To simulate delaminated areas Kingspan PIR insulation is used because it has a thermal conductivity that is very similar to that of air and thus comes closest to a real air cavity. The thermal conductivity of the PIR insulation is 0.022 (Kingspan, n.d.), the thermal conductivity of air is $0.025 + 5,5d$ where d is the thickness of the delamination in m, it is not exactly the same but the PIR insulation comes closest to real delamination and makes it possible to simulate the delamination in experiments.

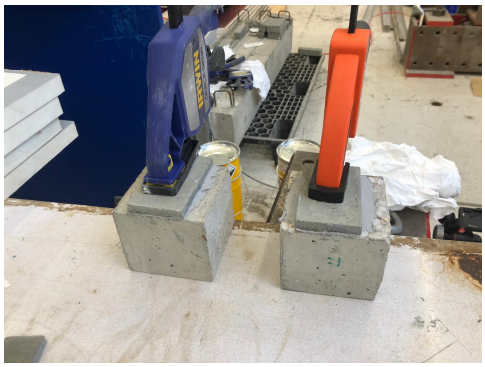
The sizes of these PIR layers are as follows: 20x20x1.05 [mm] and 30x30x1.3 [mm]. In Figure 3.1 the test samples are shown before they were glued, Figure 3.1a shows the sample with the simulated air cavity of size 20x20x1.05 [mm] and Figure 3.1b shows the sample with the simulated air cavity of size 30x30x1.3 [mm].

(a) PIR square: 20x20x1.05[mm]

(b) PIR square: 30x30x1.3[mm]

Figure 3.1: Unglued proof of concept test samples

The samples are glued together using Sikadur-30 Normal, this is a two component epoxy glue. The gluing and the glued samples are shown in figure Figure 3.2.



(a) Gluing the samples

(b) Glued samples

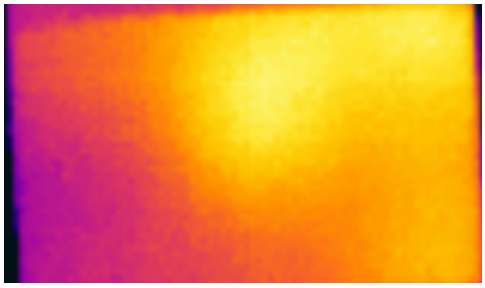
Figure 3.2: Gluing the test samples.

These samples have been used in the tests to proof the concept. In the next subsection the tests will be further explained.

3.1.2. Testing

In this stage of the research a lot of small test have been conducted. Different methods of heating up the specimen have been tried.
First it was tried to heat up the specimen with a heat gun from a distance. This resulted in a visible spot of delamination for the simulated delamination of 30x30x1.3[mm] but it did not result in a visible delamination where the PIR square of size 20x20x1.05 [mm] was used. Having the heat gun at a distance from the sample also caused an nonuniform distribution of heat, which is not desirable. The heating from a distance and the nonuniform distribution is shown in Figure 3.3

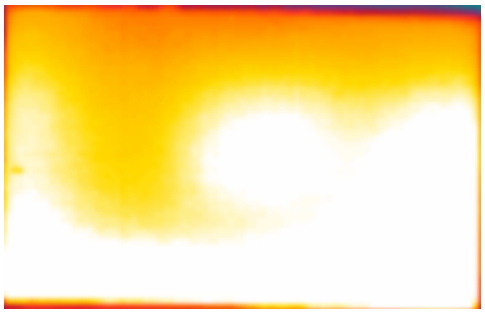
(a) Heating from a distance with a heatgun



(b) Nonuniform distribution of heat, scale: 27 °C - 32°C

Figure 3.3: Heating the specimen from a distance with a heat-gun.

Afterwards it was tried to heat up the specimen from close by and moving the heat-gun against the lamel of UHPFRC. This method also resulted in a uneven distribution of heat just after removing the heat source. However, in contrary to the heating from a distance, after a little bit of cooling time the distribution of heat became more evenly distributed. Another advantage of this method is the rapid increase of temperature at the surface of the UHPFRC lamel. Figure 3.6a shows the initial uneven distribution of heat and Figure 3.6b shows the heat distribution after a little cooling time.



(a) Initial, uneven, heat distribution, 10 seconds after heating, scale: 22.3°C - 53.2°C

(b) More even heat distribution, 100 seconds after heating, scale: 25°C - 45°C

Figure 3.4: Heat distribution with close-by heating with heat gun, scale: 30 °C - 49°C

A third method to heat up the specimen was tried with a halogen lamp of 300W, this halogen lamp was placed close to the testing surface. The initial distribution of heat was in the beginning not perfectly even, but, the same as with the heat gun, this disappeared after some initial cooling time. An advantage of the halogen lamp compared with the heat gun, is that is easier to heat up larger areas more evenly. The heat distributions can be seen in Figure 3.6 and the setup of heating the specimen and taking the picture can be seen in Figure 3.5

(a) Setup for heating the specimen with a halogen lamp.

(b) Setup for taking the picture after heating.

Figure 3.5: Heating the specimen with a halogen lamp.

(a) Initial, uneven, heat distribution, 10 seconds after heating, scale: 30°C - 43.3°C

(b) More even heat distribution, 100 seconds after heating, scale: 25°C - 35°C

Figure 3.6: Heat distribution due to heating with halogen lamp.

3.2. Main experiments

After the proof of concept experiments have been conducted, new test samples have been made to test the workings of the standardized approach and to validate the model which is made. This standardized approach is explained in chapter 6 and the model is explained in chapter 7.

3.2.1. Preparation of the test samples

Different test samples have been made to validate the model and to develop the standardized approach. A point of attention before making the test samples is the fact that the simulated delaminated areas in the proof of concept test samples were placed in the middle of the sample. When heating a surface it needs to be taken into account that edge effects are present, this causes the area in the middle of the surface to be heated slightly more then the outer edges of the surface. Therefore it is chosen that in the new test samples the delaminated areas are not simulated in the middle of the sample. This is to be certain that the increase in temperature is due to the delaminated areas and not due to edge effects. Three new test samples have been made, these are shown and explained below.

(a) Test sample with three delaminations at small distance from each other.

(b) Test sample with three delaminations at a bigger distance from each other and a test sample with two delaminated areas with thicknesses of 0.1mm made of paper and 0.5mm made of PIR.

Figure 3.7: Test samples

In Figure 3.7a one of the three test samples can be seen before gluing. This test sample has three delaminated areas which are simulated by PIR squares of 30x30mm with a thickness of 0.8mm. The squares are placed at a distance of 35mm from the edge of the NSC block, which has a size of 150x150mm. These PIR squares have first been attached to the NSC block with a little bit of vaseline, this is to prevent the PIR squares from shifting during the gluing stage.

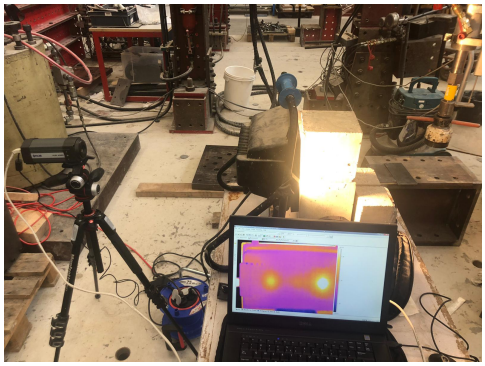
In Figure 3.7b the two other test samples can be seen before gluing. One sample has three delaminated areas which are simulated by PIR squares of 30x30mm with a thickness of 0.8mm. The squares are placed at a distance of 20mm from the edge of the NSC block, which has a size of 150x150mm. The other test sample has two delaminated areas, one is simulated with a square made out of PIR with a size of 30x30mm with a thickness of 0.5mm. The other delaminated area is simulated with a square of paper with a size of 30x30mm with a thickness of 0.1mm. Paper has a thermal conductivity of 0.05[W/mK] under normal conditions (James, 2019). These PIR and paper squares have first been attached to the NSC block with a little bit of vaseline, this is to prevent the PIR squares from shifting during the gluing stage.

3.2.2. Test setup

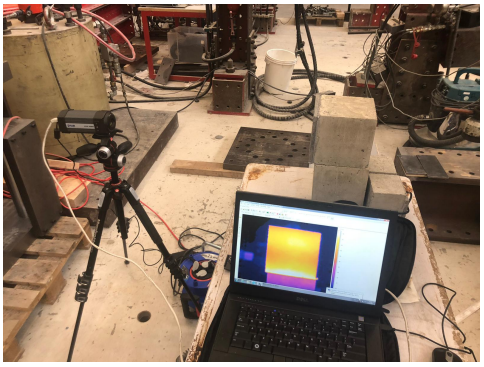
In this subsection the test setup will be briefly explained and shown. The test setup has been changed during the research, both setups will be explained.

Close-by heating

The first setup has the heating source, which is a halogen lamp of 1000W which creates 22000 lumen, directly at the surface of the UHPFRC layer. This creates a heat-flux of 10000W/m^2 at the surface. The camera was placed such that the entire specimen is in view after the lamp has been removed. The setup of the heating process, with the lamp up close is shown in Figure 3.8a. The setup of recording the cooling process is shown in Figure 3.8b



(a) Setup of heating the specimen with close-by heating with a halogen lamp.



(b) Setup of recording the cooling the specimen.

Figure 3.8: Heat distribution due to heating with halogen lamp.

The downside to close-by heating with the halogen lamp turned out to be that the surface of the specimen did not heat up evenly enough. The results from testing with this setup are presented in Appendix C and it can be seen that a more even distribution of heat is needed. Therefore it was tried to put the halogen lamp at a distance to the specimen, such that the distribution of heat at the surface was more constant. The effect of this is however, that the heat-flux decreases and thus the heating time will increase by a significant amount before the minimal delamination will be visible.

Heating from a distance

The second setup has the heating source, which is a halogen lamp of $1000W$ which creates 22000 lumen, placed at a distance of 40cm from the surface of the UHPFRC layer. This distance has been chosen due to trials with heating up a NSC block from different distances and then calculating the standard deviation during cooling of the specimen. At 40cm distance the standard deviation from the heat source itself was considerably low, which indicated that this would be a good heating distance. This manner of heating created a heat-flux of $1750W/m^2$ at the surface. The camera was placed such that the entire specimen is in view after the lamp has been removed. The setup of the heating process and the position of the camera are shown in Figure 3.9

Figure 3.9: Heating the test sample from 40cm distance with a halogen lamp.

A downside to heating from a bigger distance is the decrease in heat-flux, which means it takes longer for the RTC to increase significantly. This will be explained in more detail in chapter 6. The main advantage of this way of heating is that the distribution of heat is far more even than with the close-by heating and will give better results.

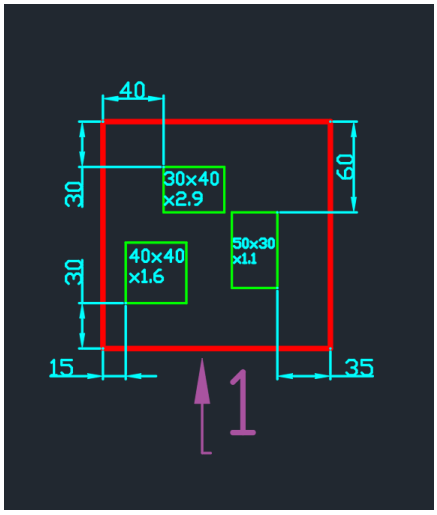
3.2.3. Gathering data

Gathering the data is done with the use of the FLIR A320 camera. With this camera a recording is made of the surface of the UHPFRC layer during the cooling time. This recording gathers the temperature at the surface for every pixel individually. This data can be extracted by using the FLIR TOOLS application on Windows, this application transforms the recording into a csv file with the temperature of every pixel per frame of the recording. This is then data which can be processed using a python script.

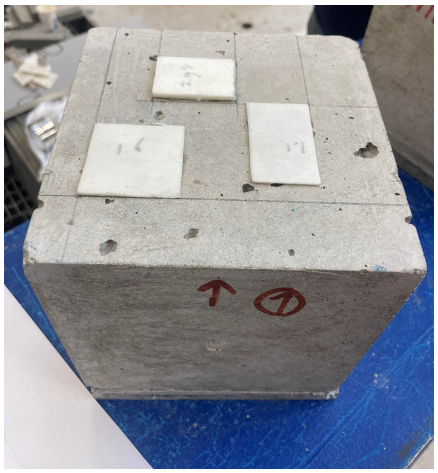
3.3. Blind experiments

The author arranged that two other people created two test samples-so that the author could not know the location or size of the delaminated areas in the sample. The testing of these samples was done with heating from a distance of 40cm, as is explained in section 3.2.

The samples are shown in Figure 3.10 and Figure 3.11.

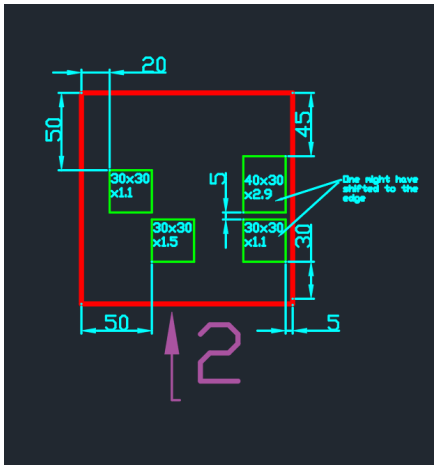


(a) Distribution of the delaminated area of the first sample presented in AutoCad.

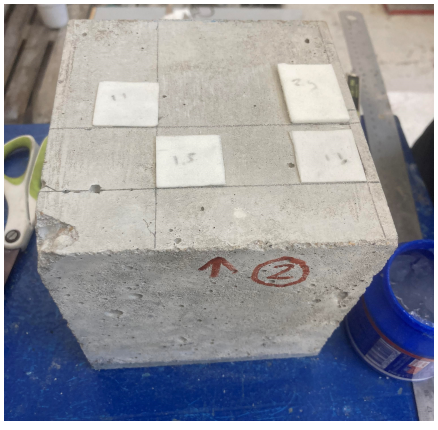


(b) Simulating the delaminated areas of the first sample with PIR.

Figure 3.10: First test sample for the unknown delaminated areas.



(a) Distribution of the delaminated area of the second sample presented in AutoCad.



(b) Simulating the delaminated areas of the second sample with PIR.

Figure 3.11: Second test sample for the unknown delaminated areas.

3.4. Real experiment

After the blind experiments have been performed, it was time to test a real sample, one where it is completely unknown where exactly the delaminated area would be. For this experiment an NSC beam with an attached layer of UHPFRC of 10mm thickness which was cast in situ was used. Even though the delaminated area was supposed to be completely unknown, this specimen had a big part of the layer of UHPFRC which was completely delaminated from the side. In Figure 3.12 this is clearly visible.

Figure 3.12: Delamination in the beam which is clearly visible.

The beam had a length 140cm and a height of 20cm and due to its size it was needed to upscale the procedure of testing. In this experiment three halogen lamps of 1000W which produced 16000 lumen each. The camera was also placed at a significant distance to the beam, so that the entire beam would be in view. After heating the table with the lamps was removed and put aside as to not disturb the recording. This setup can be seen in Figure 3.13.

(a) The setup of heating the beam with three halogen lamps.

(b) The setup of recording the beam after heating.

Figure 3.13: Setup for conducting the experiments on a beam.

Because the amount of lumen produced by these lamps was not the same as the halogen lamp which was used in the previous experiments, and because the lamps could influence each other, the light intensity was measured in Lux at the surface of the beam. The distance of the lamps to the beam was determined as follows:

The lamp produces 16000 lm, so the light which is emitted from the lamp has an energy density of 16 lm/W. To determine at which distance the lamps need to be from the beam to produce a heat-flux of 1750 W/m^2, the light intensity at the surface of the beam was measured in Lux, which equals lm/m^2.

A heat-flux of 1750 W/m^2 gives that the light intensity at the beam should be: 1750 x 16 = 28000 Lux.

The light intensity at the surface of the beam was measured with the Phyphox app, the distance of the beam was changed until the light intensity roughly equaled 28000. These trial measurements can be seen in Figure 3.14.

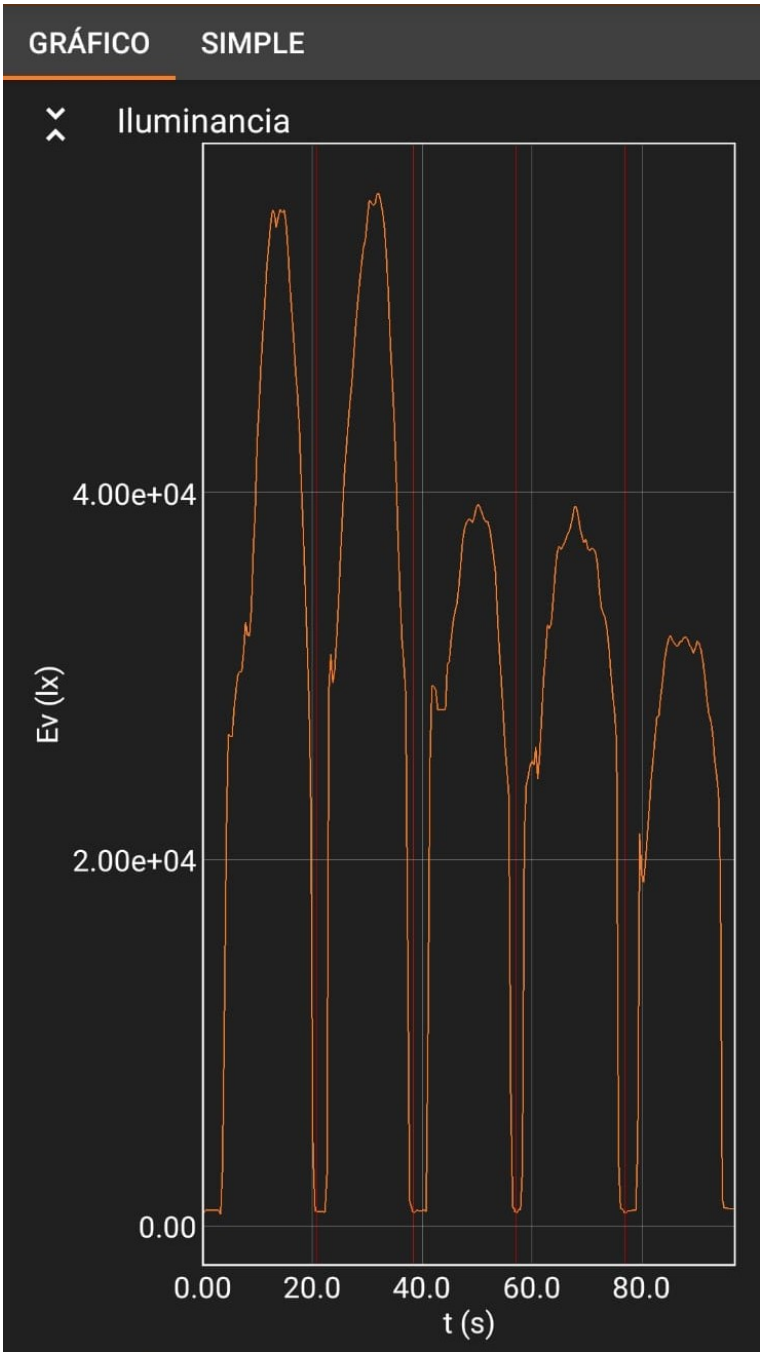


Figure 3.14: Different trials of different distances to the beam to find the right light intensity.

4

Detecting delaminated areas

When the optimal frame to be investigated is determined and extracted from the recording, which is explained in chapter 6, it is time to identify the location of the delaminated areas. The identification of the location of the delaminated areas is an important factor to be investigated in this research. In the literature review in chapter 2 three methods were mentioned to determine the locations of delaminated areas. From these methods the SNR method and the second derivative method have both been tried in the proof of concept stage in this research. These trials will be discussed in the first section of this chapter. After these trials, one method is chosen as the preferred method and will be worked out so it can be used in combination with the standardized approach explained in chapter 6.

4.1. Possible methods

In this section two possible methods for detecting delaminated areas, the signal to noise ratio (SNR) method and the second derivative method, will be explained and there results will be presented and discussed. Afterwards a comparison will be made between these methods.

4.1.1. Signal to noise ratio method

The SNR method has been tried using a csv file from the FLIR application and using a grey-scale image also conducted in the FLIR application. These approaches will both be explained in this subsection. The SNR method uses the following equation:

$$SNR = 20log_{10}\frac{|S_d - S_a|}{\sigma_{S_a}} \tag{4.1}$$

Where:
S_d = the average temperature above the delaminated area
S_a = the average temperature above the sound areas
σ_{S_a} = the standard deviation of the sound areas

First this method has been tried with a data-set gathered from the FLIR application in the form a csv file. Here, instead of the average temperature above the delaminated area, the individual temperature values per pixel was taken. In the proof of concept stage the average temperature above the sound areas has been calculated by eliminating outliers from the data-set and then taking the average of the reduced data-set. Processing the data that gives the result shown in Figure 4.1, gives that the amount of outliers there was 8.85%

From this reduced data-set the standard deviation (STD) was taken. Another thing that is changed compared to Equation 4.1 is that not the absolute value of $S_d - S_a$ is taken but also the negative values are taken, this is done because delaminated areas have a higher temperature and thus $S_d - S_a$ gives a positive value if it is a delamination, if it gives a negative value this would be due to noise which is not of interest here. These changes give the following equation:

$$SNR = 20log_{10}\frac{T_d - S_a}{\sigma_{S_a}} \tag{4.2}$$

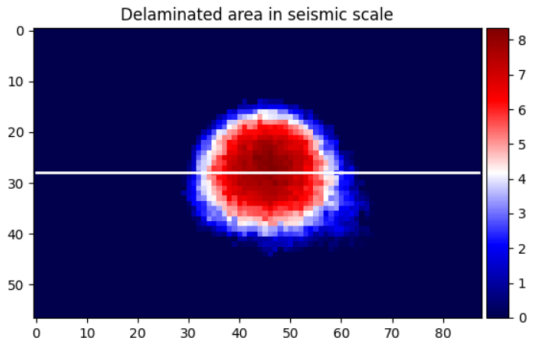
Where:
T_d = the temperature per pixel
S_a = the average temperature above the sound areas
σ_{S_a} = the standard deviation of the sound areas

Equation 4.2 gives the SNR value per pixel, if this value is below zero it means that the difference in temperature is so small that no delamination is expected, if this value is above zero, it means that such a change in temperature is found that a delamination is expected.

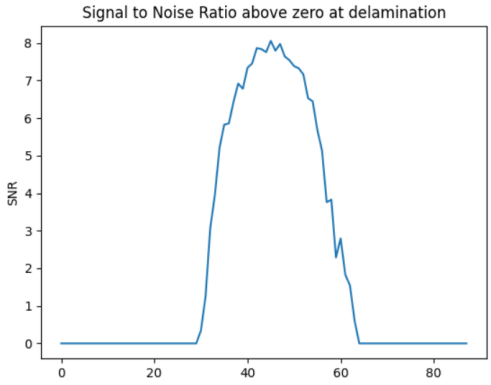
Another way to use the SNR method is to use a picture from the FLIR data in a grey-scale. This works exactly the same as the method described above, only now instead of using the temperature of every pixel, the value of the grey-scale of every pixel is used. An advantage here is that the data does not have to be extracted to a csv file and that the size of the image is bigger than the size of the data-set, meaning that one picture holds more pixels and gives a more pleasant picture. A downside to using grey-scale images however, is that the scale settings is done subjectively instead of objectively, which is unwanted in this study. Another downside is the fact that due to the fact that more pixels are presented, it could be thought that this holds more data. The fact is however that these pixels are interpolated from the original pixels given in the csv file, which means that a lot of the data in the grey-scale image is fictitious.

The choice has been made that for this method the use of the data-set extracted from the FLIR application in the form of a csv file is the most accurate and objective, as it gives the exact temperature of a pixel. The advantage of the higher resolution given by a picture does not exceed the advantage of the objective quality of the csv data. Therefore the results shown below are only those of csv file data. The results computed with the grey-scale image data can be found in Appendix A

The results of this method are shown in Figure 4.1. These are the results from an experiment with a sample of 10x6cm with a delamination of 30x30x1.3mm. In Figure 4.1a the positive values of the SNR are plotted in a seismic scale over the area of the specimen. In Figure 4.1b the positive values of the SNR on the mid-line of the specimen are plotted. Finally in Figure 4.2 the known delaminated area is plotted with a white box over the results of the SNR method. As can be seen the delaminated area calculated with the SNR method almost covers the entire white box, this indicates that this method works well.



(a) Delaminated area, values of the SNR above zero shown.



(b) SNR values above zero over the mid-line of the delaminated area.

Figure 4.1: Delamination detection with csv file data, using the SNR method. Plotting the positive SNR values.

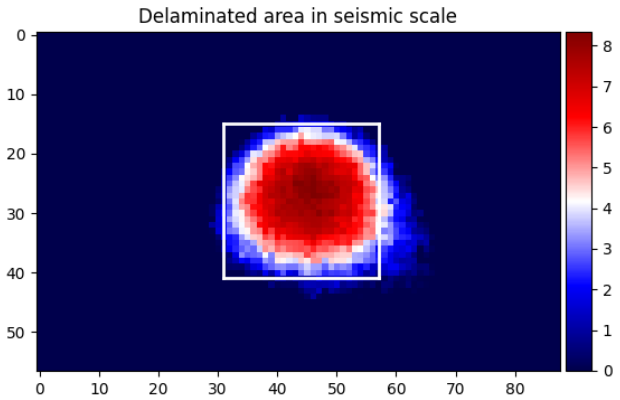


Figure 4.2: Delaminated area, values of the SNR above zero shown, with a white box which indicates the known delaminated area.

4.1.2. Second derivative method

The second derivative method is an entirely different method to detect delaminated areas. In this method a 4^{th} order polynomial is fit through every row and every column of the csv data-set provided by the FLIR application. From every polynomial the second derivative is calculated and the points are detected where this second derivative is zero. These points indicate inflection points and would, according to theory, indicate where a delaminated area begins. This is also done for the grey-scale image extracted from the FLIR Tools application. The results of the csv file data are shown in Figure 4.3 and the results of the grey-scale image data are again found in Appendix A. It can be seen that the inflection points show boundaries of a delaminated area in the middle of the specimen, the top boundary, however, is not visible. The indicated delaminated area has the right position, however the size is not entirely right. The script which is created to perform the second derivative method this figure is found in **??**

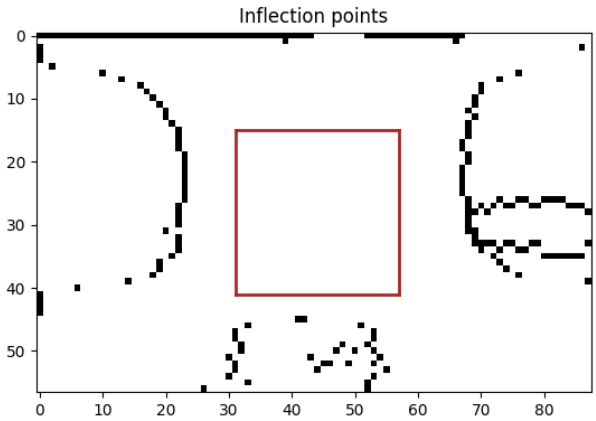


Figure 4.3: Inflection points of csv data-set with a brown box at the known delaminated area from the second derivative method.

4.2. Conclusion

Two different methods have been explained for detecting delaminated areas. The SNR method makes use of the temperature of every pixel and compares this to the average temperature and standard deviation of a sound area. The second derivative method makes use of the temperature of every pixel and fits a curve through these values to determine the inflection points. Both methods have been used to detect a delaminated area of 30x30x1.3mm and the results have been compared to the actual delaminated area. Regarding the effectiveness of the two methods the following can be concluded:

- The SNR method is more accurate about the surface area of the delaminated area than the second derivative method.
- The second derivative method overestimates the area of the delamination.
- The SNR method is easier to interpret than the second derivative method.

Finally it is decided to proceed with the SNR method.

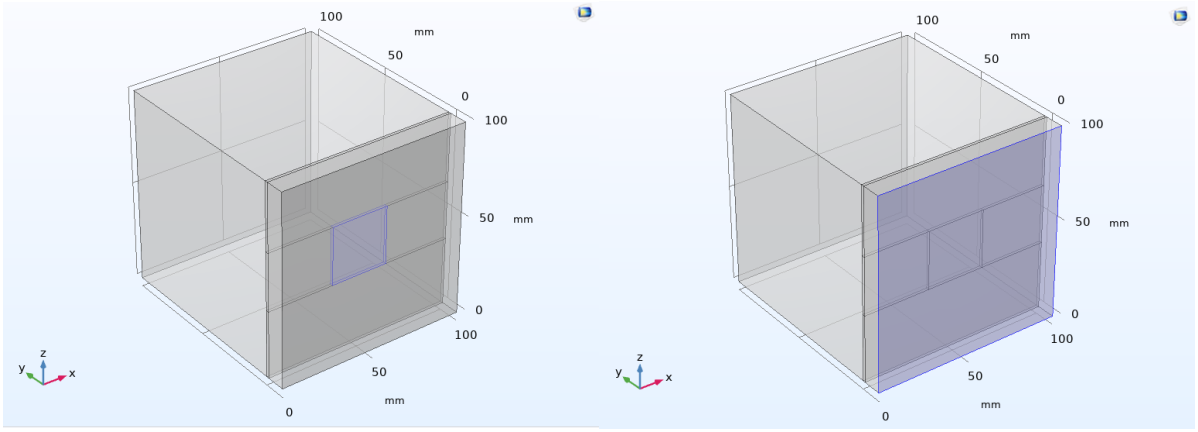
5

Numerical modelling in COMSOL

In this chapter the use of COMSOl as an FE model will be explained and the data gathered from COMSOL will be validated with experimental data.

5.1. Simulations

In this section the simulation of different delaminations in COMSOL will be explained and shown. A NSC block is made and a 'block' of air 30x30mm and a certain thickness is constructed to the side of the NSC block. The sides around the air block are filled up with NSC blocks of the same thickness as the air block. At last a block of UHPFRC of 10mm thickness is attached on top of the air block. This is the sample which is used in COMSOL to gather all the required information. The sample in COMSOL is shown in Figure 5.1a.



(a) Simulation of delaminated area in COMSOL, air in blue

(b) Simulation of the heat-flux in COMSOL, heat-flux area in blue

Figure 5.1: Simulation of experiments in COMSOL

As heat-source a heat-flux is applied on the surface of the UHPFRC layer, which can be seen in Figure 5.1b.

Gathering the temperature data from COMSOL was done with 2 lines to the side of the delaminated area and one line above the delaminated area, over these lines the temperature is gathered in a csv file to process in the analytical model.

5.2. Validation

To validate the data gathered from COMSOL a small experiment was conducted and the results from this experiment have been compared to the simulations made in COMSOL. The experiment and the

comparison of data is discussed in this section.

This validation process is done both for heating the specimen for 120 seconds with a heat-flux of $10000 W/m^2$ at the surface of the UHPFRC layer and for heating the specimen for 1500 seconds with a heat-flux of $1750 W/m^2$ at the surface of the UHPFRC layer. This is done due to the fact that the method of heating has been changed throughout the research, because the heating with a heat-flux of $10000 W/m^2$ did not give an even enough distribution of heat.

In chapter 6 the choice for these heating times is made and explained.

The experiment that was conducted was to heat up the proof of concept specimen with the delaminated area of 30x30mm and a thickness of 1.3mm. After heating, the heat source was removed and a recording was made. Temperature data was gathered from the recording above non-delaminated area and above the delaminated area.

The COMSOL data was gathered from a simulation of a delaminated area made of PIR with a size of 30x30mm with a thickness of 1.3mm. The temperature data gathered from COMSOL was from both the non-delaminated area as well as the delaminated area.

All this data which is gathered is used to compare the temperature above the delaminated area and the non-delaminated area which are calculated by COMSOL and which are measured in the experiment. This data is also used to calculate the SNR and the RTC. The process of calculating the SNR and RTC values is schematized in Figure 5.2

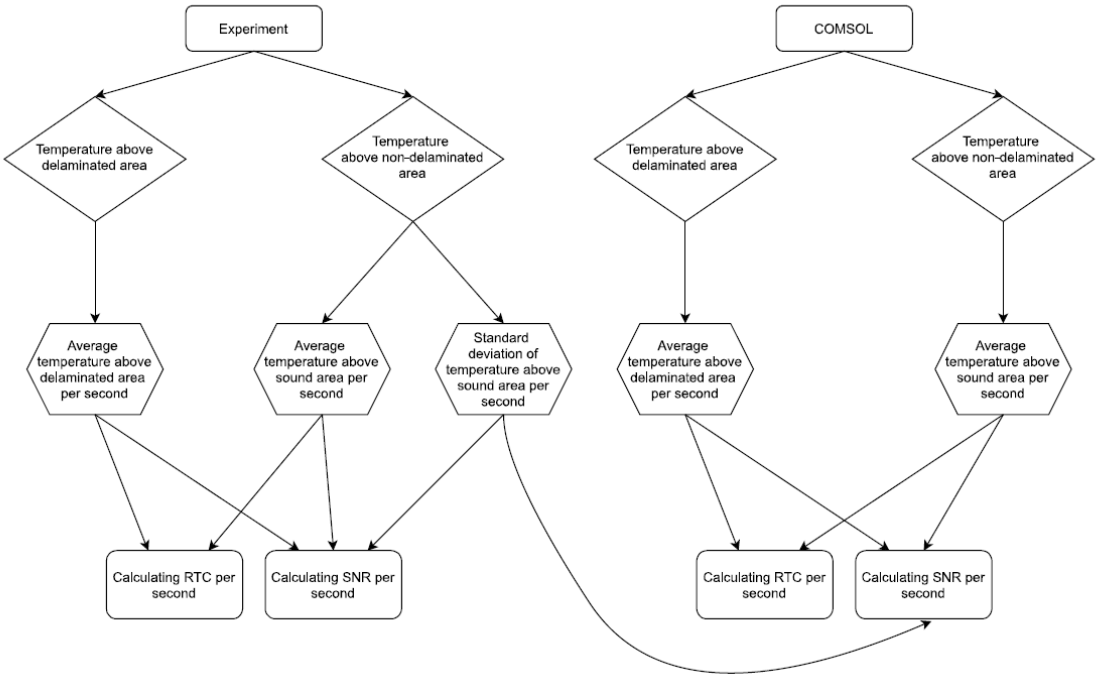


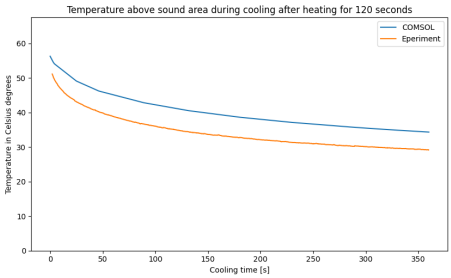
Figure 5.2: Flowchart: How the SNR and RTC is calculated from both COMSOL data and experiment data

Figure 5.3 shows graphs of the above mentioned values during the cooling time after heating the specimen for 120 seconds with a heat-flux of $10000 W/m^2$. These values have been calculated with data gathered from COMSOL and with data gathered from the experiment conducted in the lab.
Figure 5.4 shows graphs of the above mentioned values during the cooling time after heating the specimen for 1500 seconds with a heat-flux of $1750 W/m^2$. These values have been calculated with data gathered from COMSOL and with data gathered from the experiment conducted in the lab.

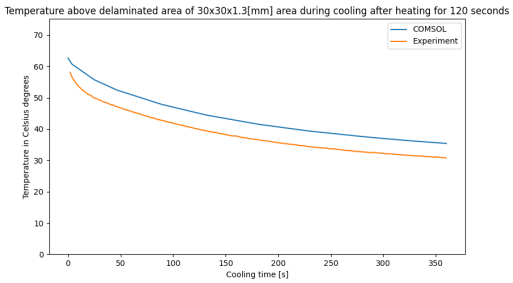
As can be seen in the graphs in Figure 5.3 and Figure 5.4 the temperature above the sound and delaminated areas are both overestimated in the COMSOL calculation. The absolute temperature difference is however already better estimated, especially in the simulation of a heat-flux of $1750 W/m^2$

with a heating time of 1500 seconds. The most important value is the SNR, because this is the value which is used to determine the optimal cooling time, or in other words, the optimal frame. As can be seen the SNR calculated with the temperature difference from COMSOL is either underestimated, in the case of a heat-flux of 10000W/m^2 with a heating time of 120 seconds, or actually overlaps very well with the actual SNR from the experiment, as is the case for a heat-flux of 1750W/m^2 with a heating time of 1500 seconds. The RTC is also underestimated by the simulation in COMSOL, however, the shape of the graph is good enough to not rise concerns about the accuracy of the COMSOL simulation.

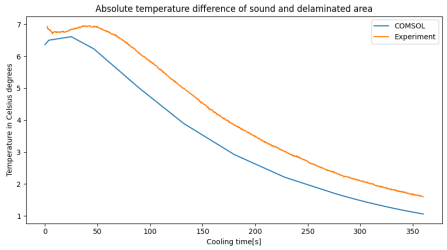
Due to these facts, it can be stated that, although the simulation in COMSOL does not portray reality perfectly, it still can be used for the purpose of this research.



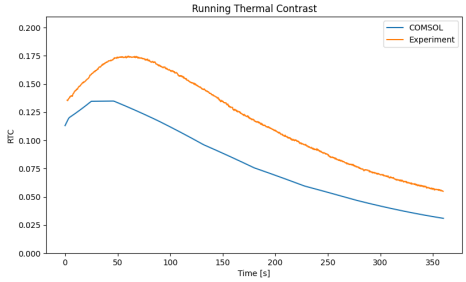
(a) Temperature of sound area over time calculated with data from COMSOL and from the experiment.



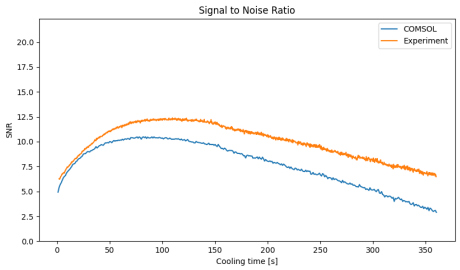
(b) Temperature of delaminated area over time calculated with data from COMSOL and from the experiment.



(c) Absolute difference in temperature above delaminated area and sound area calculated with data from COMSOL and from the experiment.

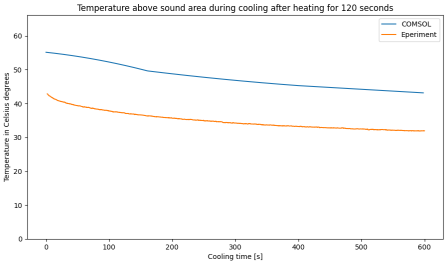


(d) RTC over time calculated with data from COMSOL and from the experiment.

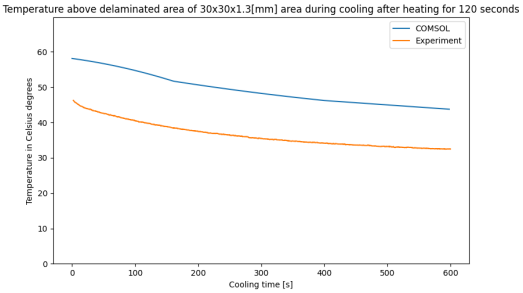


(e) SNR over time calculated with data from COMSOL and from the experiment.

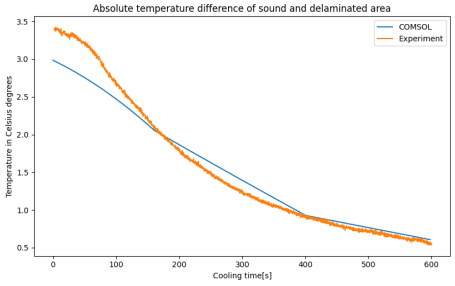
Figure 5.3: Verification of values from data gathered from COMSOL and from experiment of heating for 120 seconds with a heat-flux of 10000W/m^2



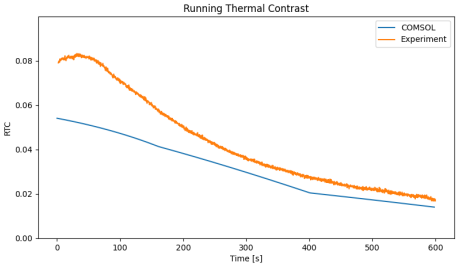
(a) Temperature of sound area over time calculated with data from COMSOL and from the experiment.



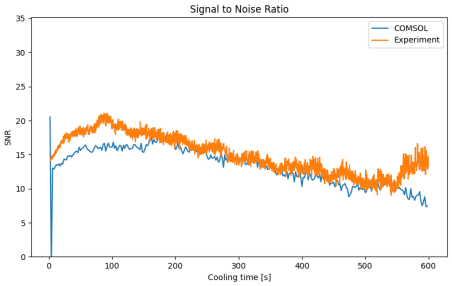
(b) Temperature of delaminated area over time calculated with data from COMSOL and from the experiment.



(c) Absolute difference in temperature above delaminated area and sound area calculated with data from COMSOL and from the experiment.



(d) RTC over time calculated with data from COMSOL and from the experiment.



(e) SNR over time calculated with data from COMSOL and from the experiment.

Figure 5.4: Verification of values from data gathered from COMSOL and from experiment of heating for 1500 seconds with a heat-flux of 1750W/m^2

6

Standardized approach

One of the aims of this research was to find a standardized approach to detect delaminated areas in the bond between NSC and UHPFRC with active IRT imaging. From the proof of concept experimental results, of detecting delaminated areas, the following dependency is found:

- Heating time of the specimen.
- Cooling time of the specimen.
- Environmental conditions.

In this chapter, a choice is also made for the minimal delamination for this standardized approach. After this choice is made the factors mentioned above and their influence on the standardized approach will be further explained in the above given order.

6.1. Minimal delamination

A choice needs to be made for the minimal delamination. This is dependent on the application of the specimen that is to be researched. A delaminated area of the size of 30x30x0.05mm would have a significant enough effect on the added capacity created by using a UHPFRC panel to increase the shear strength of a NSC beam. Therefore it is chosen to use a delaminated area of 30x30x0.05mm as the minimal delamination for this research.

6.2. Heating time of the specimen

In the literature it was suggested that for heating the specimen an optimal heating time would exist.

The method to determine the optimal heating time, suggested by the literature, was to calculate the RTC for multiple time intervals to determine the time where the RTC reaches its maximum. This method assumes that there is a time where the RTC reaches a maximum, this is the case when a certain temperature is prescribed as heat source. In this research, however, not a certain temperature but a certain heat-flux is prescribed. Due to this fact the RTC does not reach a maximum, it will keep increasing and approaches an asymptote. It is possible in this case to calculate the point where the rate in which the RTC increases has decreased a certain amount. This time can be obtained by writing a script that uses the average temperature of the sound area and the temperature above a delaminated area.

The term RTC has been explained in chapter 2 already and is defined as follows:

$$\Delta T_r = \frac{(T_{def} - T_{non-def})}{T_{non-def}} \tag{6.1}$$

Where:
T_{def} = The temperature above a delaminated area.

$T_{non-def}$ = The temperature above a sound area.

In COMSOL a simulation is done to determine the rate of increase of the RTC for different delaminated areas. The following sizes of delaminated areas have been simulated:

- 30x30x0.05mm
- 30x30x0.1mm
- 30x30x0.5mm
- 30x30x0.8mm
- 30x30x1.0mm
- 30x30x1.3mm
- 30x30x1.5mm

This simulation has been performed with both a heat-flux of 1750[W/m^2]. The calculation of the heating time of the heat-flux of 1750[W/m^2] is explained here below.

In Appendix B the heating time for heating with a heat-flux of 10000[W/m^2] is calculated in a bit of a different way for delaminated areas of the following sizes:

- 30x30x0.1mm
- 30x30x0.5mm
- 30x30x0.8mm
- 30x30x1.0mm
- 30x30x1.3mm
- 30x30x1.5mm

The minimal delamination has a thickness of 0.05mm, so the size which will be used to model in COMSOL is a delamination of 30x30mm with a thickness 0.05mm.

The way in which the heating time is calculated is to take the time where the RTC at the backside of the UHPFRC layer almost reaches its asymptote. The RTC is calculated at the backside of the UHPFRC layer due to the fact that while heating, the surface temperature in the COMSOL model of the UHPFRC layer is almost constant over the entire area, regardless of the delaminated area present. The optimal heating time is taken at the time where the RTC almost reaches the asymptote because at that point the relative temperature difference will not increase by a significant amount anymore.

This point is calculated as the point where the slope of the RTC reaches 0.5% of the maximum slope. This point is calculated for both a delamination of PIR and air and is found at a time of 1205 seconds and 1196 seconds respectively. It is chosen to round the heating time up to 25 minutes, which equals 1500 seconds, because this is an easier to remember amount of time and is easier to monitor.

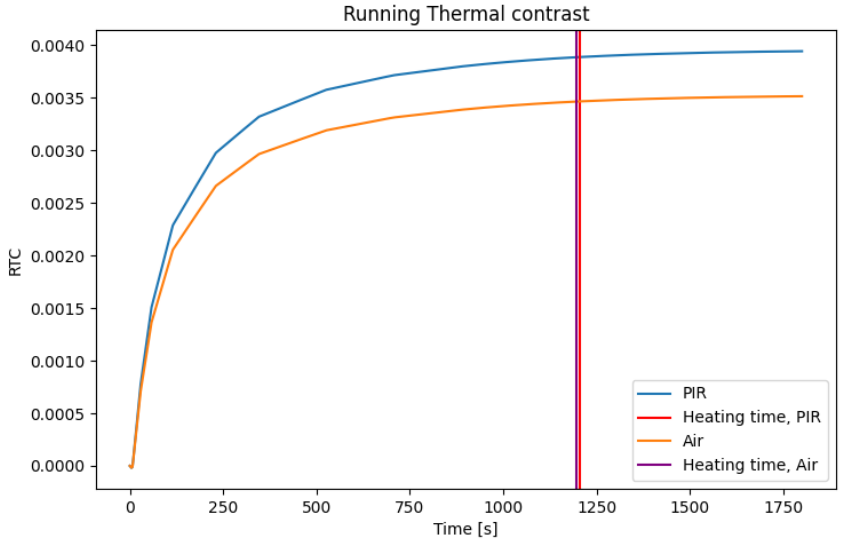


Figure 6.1: RTC at the backside of the layer of UHPFRC due to a delamination of PIR and air with the heating time

6.3. Cooling time of the specimen

Apart from the heating time, the cooling time also plays a crucial role in the ability to detect delaminated areas. At first, cooling the specimen increases the RTC and SNR of the delaminated area but after a certain time the cooling process decreases the RTC and SNR of the delaminated area. It is important to find the time where the delaminated areas are best detectable. Seeing as the smallest and thinnest delamination will also have the smallest increase in temperature above the delamination, this delamination will be governing.

A time needs to be found for when the noise of a sound area is significantly smaller than the increase in temperature above the delamination and at which this difference in temperature is optimal. This is done with the use of the previously mentioned SNR method. Why this is done with the SNR method is explained in more detail in chapter 4. The SNR method uses the following equation:

$$SNR = 20log_{10}\frac{|S_d - S_a|}{\sigma_{S_a}} \tag{6.2}$$

Where:
S_d = the average temperature above the delaminated area
S_a = the average temperature above the sound areas
σ_{S_a} = the standard deviation of the sound areas

The time which gives the optimal frame for detecting the minimal delamination can be approached in two ways. One approach can be used when one has the knowledge of a sound area in the specimen, the other approach is for when there is no knowledge about a sound area. Both approaches are explained below.

6.3.1. Situation where there is a known sound area within the specimen.

A recording is made with an IRT camera of the specimen after heating, when it is cooling down. From this recording two csv files are extracted of two areas with the temperatures of the pixels per frame. One file contains the temperatures above a known sound area and another file contains the temperatures of the entire specimen. The file of the sound area is then processed to find the standard deviation (STD) per frame, this is considered to be the STD of the sound area. The average temperature above the delaminated area is taken from the simulation in COMSOL of the smallest delamination that is needed to be detected, the average temperature above the sound area is also found with the simulation in COMSOL. More about these simulations can be read in chapter 5

Because these values are time dependent, the SNR during cooling time is calculated per second. After these are calculated, the time is found for when the SNR reaches its maximum value. This is the optimal time to detect the minimal delamination and because this delamination is governing, it means that at that point the bigger delaminations will also be detectable. After this time is determined, the frame that is linked to this certain time is extracted from the other csv file with the entire specimen. This frame is the optimal frame to investigate.

6.3.2. Situation where there is no known sound area within the specimen.

When there is no known sound area in the specimen another approach needs to be made. In this subsection a suggestion is made for how this could be done, a thing to note however, is that in this case it is not possible to have a major area of delaminated area within the specimen. The exact maximum amount of allowable delamination and the thickness and size of the allowable delamination is something that needs to be researched in a following research, this is only a suggestion of a method. It is found that this suggestion works with the amount of delamination which is simulated in this research.

In this approach a recording is made with an IRT camera of the specimen after heating, when it is cooling down. From this recording one csv file is taken, this file contains the temperatures of the pixels over the entire specimen, per frame. From this file the STD and average of the temperature is calculated per frame. With these values the optimal frame is appointed in the same way as explained in subsection 6.3.1.

6.3.3. Environmental conditions

In both cases, with or without a known sound area, the environmental conditions have already been taken into account due to the fact that the STD is taken from the recording itself, this is the value where the environmental conditions will have the highest impact. The cooling time is dependent on this STD, thus with this method of cooling, the environmental conditions have been taken into account when determining the optimal cooling time.

7

Analytical model

In chapter 4 the method of detecting the delaminated areas is chosen and explained. In chapter 6 the standardized approach of heating and cooling the specimen is explained, where for the cooling a distinction is made between a sample with a known sound area and without a known sound area. This chapter will combine both chapters into a model, this model will process the recording, which is made according to the standardized approach, by using the SNR method of detecting the delaminated areas. This chapter is divided into two sections, section 7.1 will briefly discuss the process of extracting the frame from the recording which belongs to the optimal cooling time and section 7.2 will discuss the process of using the SNR method to determine the delaminated area, this section is divided in two subsections. subsection 7.2.1 discusses the method for determining the delaminated area of a sample where there is a known sound area. subsection 7.2.2 discusses a suggested method for determining the delaminated area of a sample where there is no known sound area.

7.1. Extracting the optimal frame from the recording

Extracting the optimal frame from the recording is an important part of the model that has been created. The optimal frame is chosen in such a way that the temperature difference, at that certain time that is calculated from the COMSOL model with the minimal delamination, together with the measured STD of the sound area, at that certain time, creates the maximum value of the SNR. The way this is calculated for the situation where there is a known sound area is shown in a flowchart in Figure 7.1. The way this is calculated for the situation where there is no known sound area is shown in Figure 7.2.

7.2. Determining delaminated area

Determining the delaminated area is done with the SNR method, why this is done is explained in chapter 4. In this section two ways of implementing this method are discussed, one for a sample with a known delaminated area and one for a sample without a known delaminated area.

7.2.1. Situation where there is a known sound area within the specimen.

In this subsection the method for detecting delaminated areas for a where there is a known sound area in the specimen will be discussed.
After the optimal frame is determined via the method explained in Figure 7.3 the average temperature of the sound area is calculated together with the STD of the sound area. With these values the SNR is calculated for every pixel in the frame of the entire specimen. The positive values are stored and where a pixel has a negative SNR value the pixel will get a value of zero instead, the positive values indicate a delamination. After assigning these values to all pixels the frame is plotted in a seismic scale to show the delaminated area in a clear manner, the darker red parts indicate the centre of the delaminated area and the white and light blue parts indicate that the boundary of the delaminated area is close. The way the delaminated areas are determined in the model in this case is also shown in a flowchart in Figure 7.3.

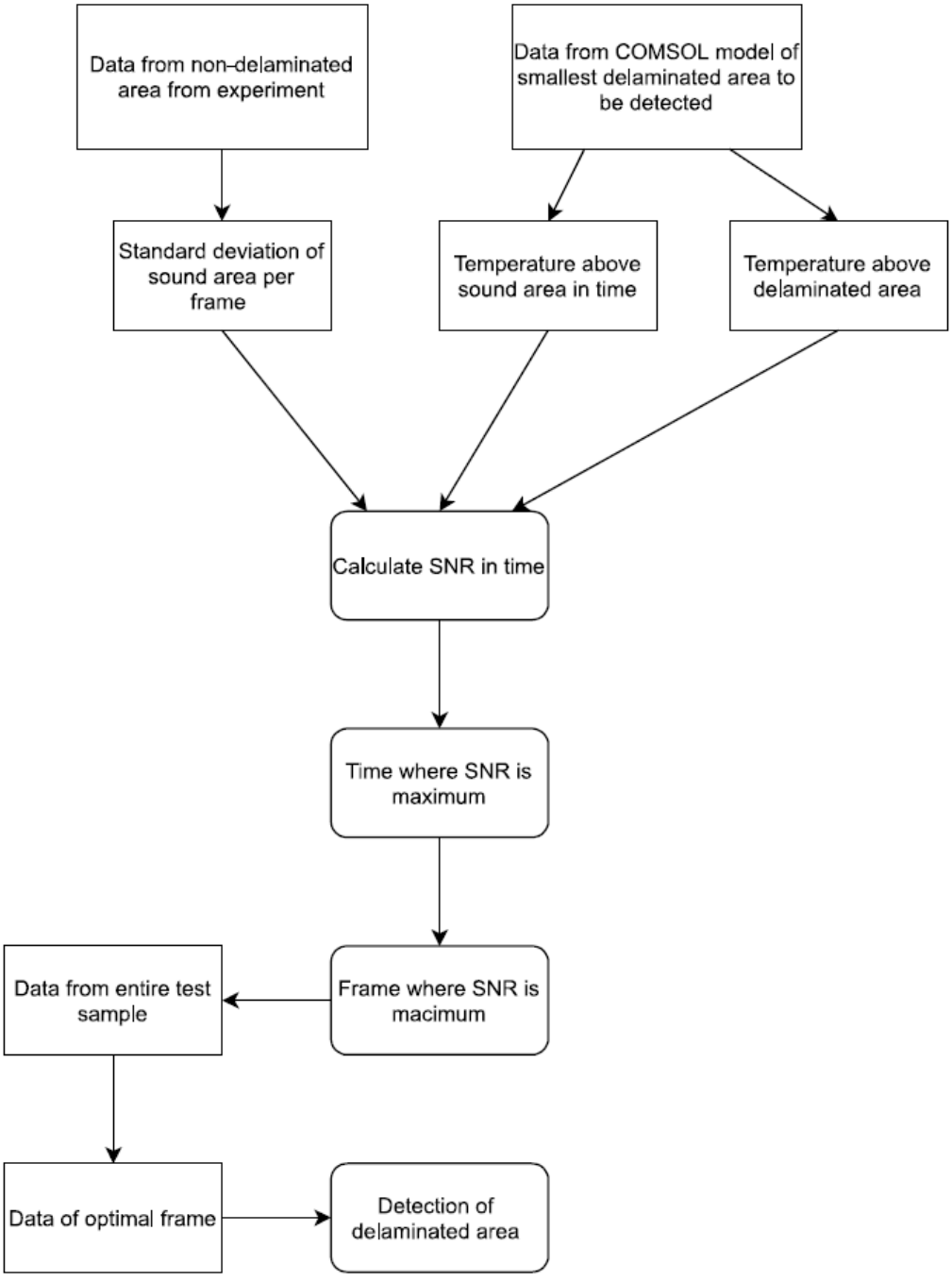


Figure 7.1: Flowchart of determining the optimal frame from a recording where there is a known sound area.

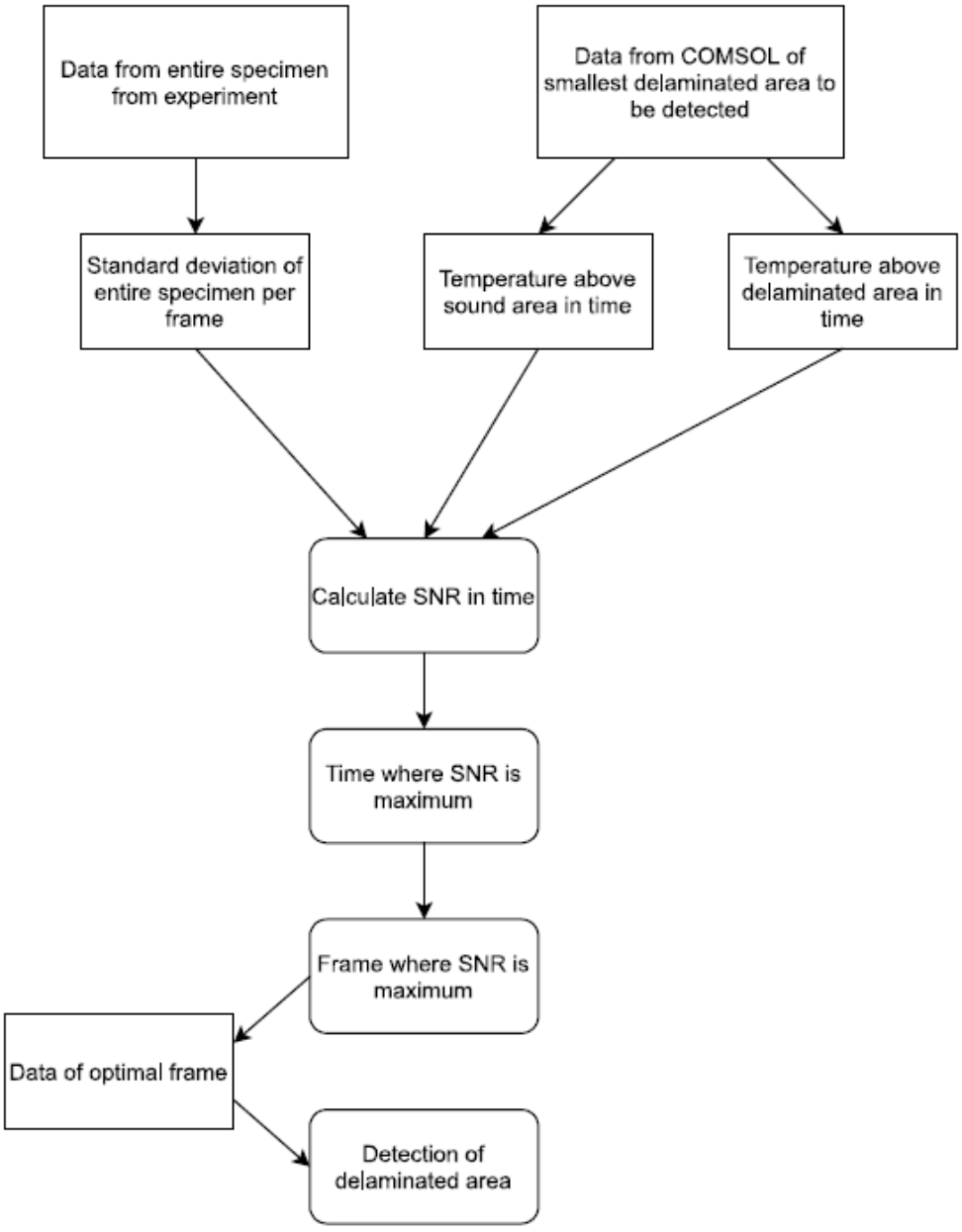


Figure 7.2: Flowchart of determining the optimal frame from recording where there is no known sound area.

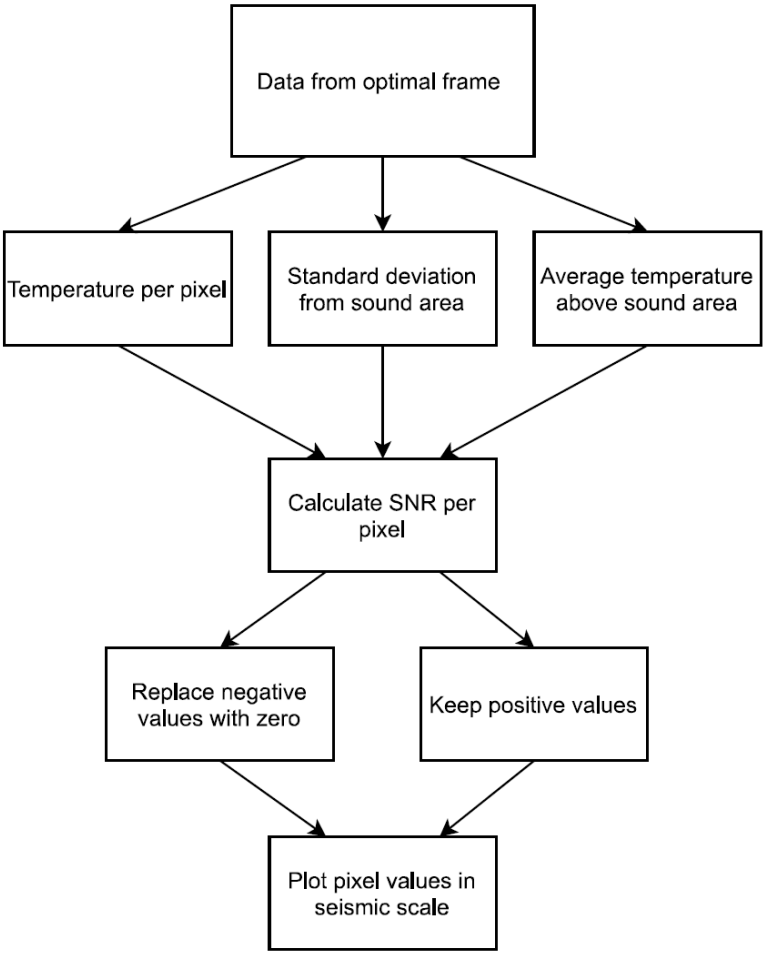


Figure 7.3: Flowchart of determining delaminated areas from the data of the optimal frame when there is a known sound area

7.2.2. Situation where there is no known sound area within the specimen.

In this subsection a method for detecting delaminated areas for a situation where there is no known sound area in the specimen will be discussed. Note that this is a suggested method which needs to be to be verified more thoroughly in a following research.

After the optimal frame is determined via the method explained in Figure 7.4 the absolute temperature difference from the simulation of the minimal delamination in COMSOL is taken at the time which matches the optimal frame. The average temperature of the entire frame is calculated and then the temperature difference is calculated for every pixel as the difference between the temperature of the pixel and the average temperature of the entire frame. This temperature difference is then compared to the temperature difference taken from the COMSOL simulation. If the temperature difference of a pixel is smaller then 50% of the temperature difference calculated by COMSOL then the pixel is seen as a sound area. The 50% is a chosen value with a high safety factor. This value needs to be lower than a 100% because the average of the entire specimen is higher then the average temperature of the sound area, which is used in the difference calculated with COMSOL. This value is a suggestion and if this method is to be researched further, it should be researched how much lower than a 100% this value should actually be.

This check is done for every pixel in the frame and in this manner a sound area is determined. From this sound area the STD and average temperature is calculated and then used to calculate the SNR of every pixel in the frame. Again the positive values are stored and where a pixel has a negative SNR value the pixel will get a value of zero instead, the positive values indicate a delamination. After assigning these values to all pixels the frame is plotted in a seismic scale to show the delaminated area in a clear manner, the darker red parts indicate the centre of the delaminated area and the white and light blue parts indicate that the boundary of the delaminated area is close. The way the delaminated areas are determined in the model in this case is also shown in a flowchart in Figure 7.4.

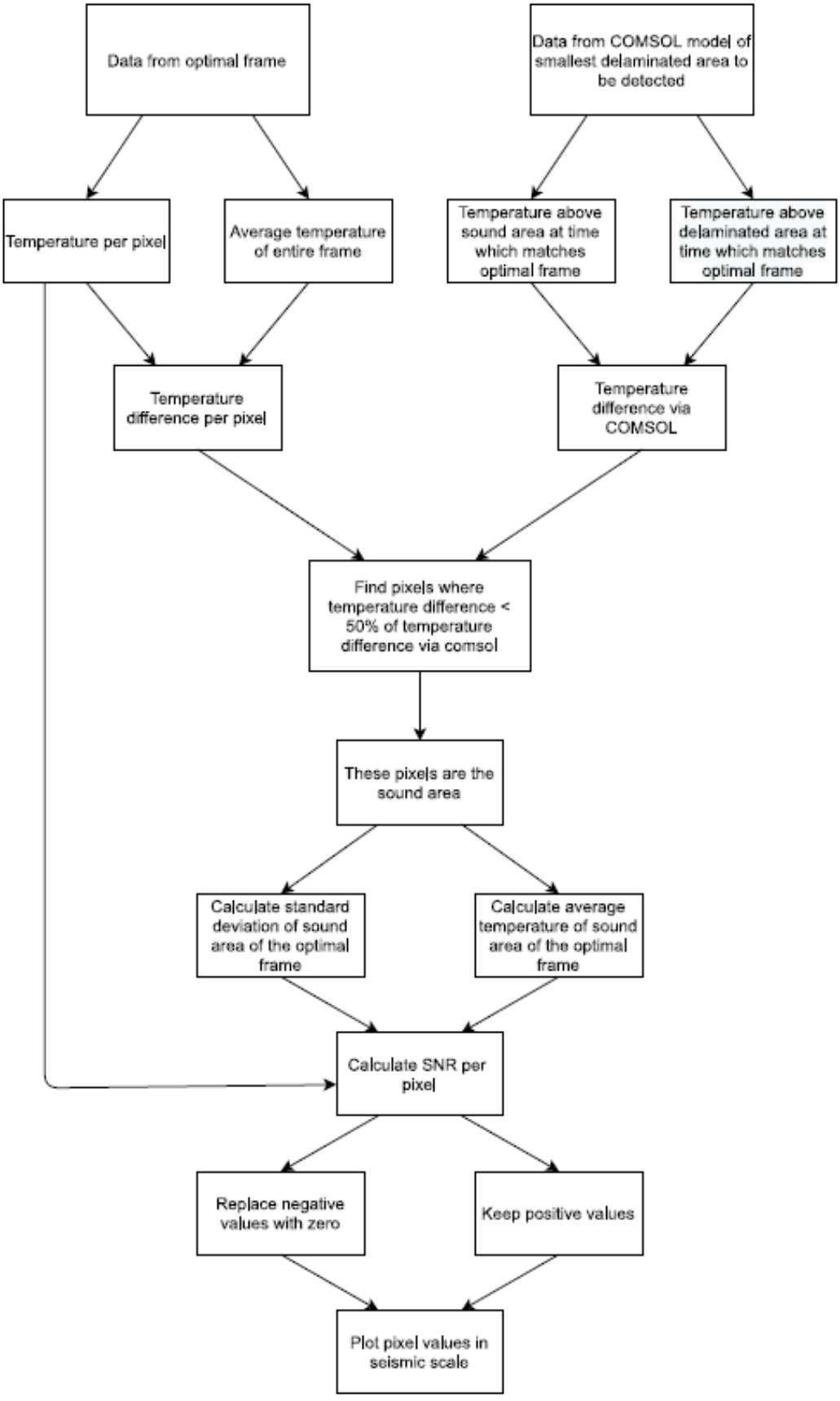


Figure 7.4: Flowchart of determining delaminated areas from the data of the optimal frame when there is no known sound area

8

Results

This chapter will discuss the results produced by the model explained in chapter 7 from the data gathered from the experiments performed by using the standardized approach, which is explained in chapter 6. Only the results from the heating with a heat-flux of 1750W/m^2 with a heating time of 1500 seconds is discussed in this chapter. The results from the heating with a heat-flux of 10000W/m^2 with a heating time of 120 seconds is discussed in Appendix C.

Two kinds of regulated experiments have been conducted:

- The main experiments; the author knew the locations and sizes of the simulated delaminations.
- The blind experiments; the author did not know the locations and sizes of the simulated delaminations.

For both the results will be given below in the form of a seismic scale picture and the following ratios will be given as well:

- true positive ratio which is calculated as follows: $TPR = \frac{TP}{TP+FN}$
- true negative ratio which is calculated as follows: $TNR = \frac{TN}{TN+FP}$
- false positive ratio which is calculated as follows: $FPR = \frac{FP}{FP+TN}$
- false negative ratio which is calculated as follows: $FNR = \frac{FN}{FN+TP}$

The main experiments were conducted with three different samples, for all three samples the results will be shown in this chapter. For the blind experiments two samples were used, the results of both of them will be presented in this chapter, including the delaminated area, which was presented to the author after processing the data.

A third kind of experiment was conducted as well. This was done with a beam where there was a part of delamination known, however the exact starting location of the delaminated area was unknown. More about this experiment sample can be found in chapter 3. The results for this experiment is given in section 8.3 in the form of a seismic scale picture of the specimen and three graphs of the SNR values over lines plotted along the longitudinal axis of the beam.

8.1. Main experiments

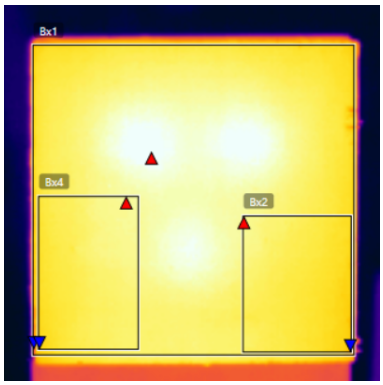
The experiment was to heat up the three specimens with a heat-flux of 1750W/m^2 for 1500 seconds and let it cool down while recording. The data from the recordings have been gathered. From the three specimen a csv file with the temperatures per pixel of every frame over the entire specimen has been extracted as well as a csv file with the temperatures per pixel of every frame over a sound area. For every specimen two sound areas have been used and the results from both are displayed as well. In the FLIR application three boxes are drawn from which the data is gathered. Bx1 is the box with the

data of the entire specimen, the other boxes are the boxes above sound areas. As well as using the method for a known sound area, the method for detecting the delaminated area for the unknown sound area is also used on these samples.

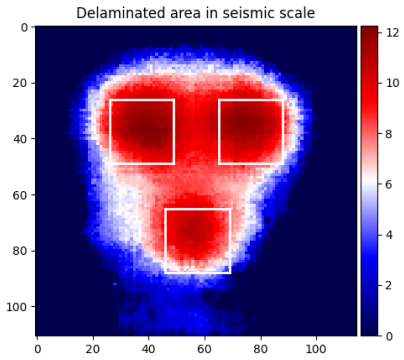
For all three samples the results, in the form of a seismic scale picture, are presented, the reference values of the sound areas are presented and in addition to these values the probability density functions of the different sound areas per sample are shown.

The ratios to determine the accuracy of the model are presented in Table 8.2, Table 8.4 and Table 8.6 for all three samples and for all three methods. Note that the FPR is very high, because this method takes all pixels with a value of the SNR above zero as a delaminated area, however, as can be seen in the results below, the light blue and white pixels indicate the boundary of the delaminated area.

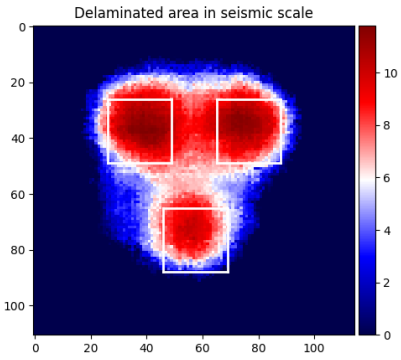
8.1.1. Specimen with three delaminations of 30x30x0.8mm of pir at a distance of 35mm of the outer sides.



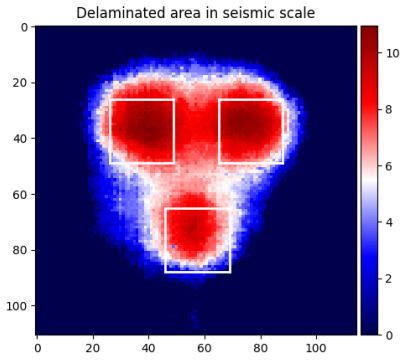
(a) Absolute difference in temperature above delaminated area and sound area calculated with data from COMSOL and from the experiment.



(b) Result computed with the the data of the sound area of Bx2.



(c) Result computed with the the data of the sound area of Bx4.



(d) Result computed with the suggested method for a specimen without a known sound area.

Figure 8.1: Results from the specimen with three delaminations of 30x30x0.8mm of pir at a distance of 35mm of the outer sides.

	Average temperature [°C]	Median [°C]	Standard deviation [°C]
First sound area	41.30	41.38	0.6488
Second sound area	41.79	41.86	0.5554
Unknown sound area	41.19	41.32	0.7066

Table 8.1: Reference values of the sound areas used for the sample with three delaminated areas at a distance of 35mm from outer edge.

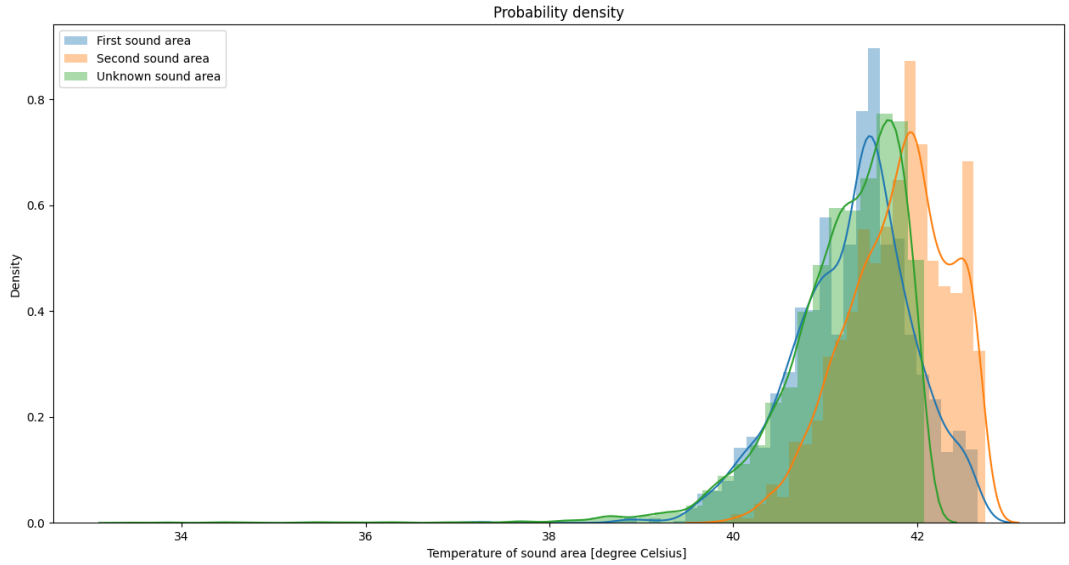
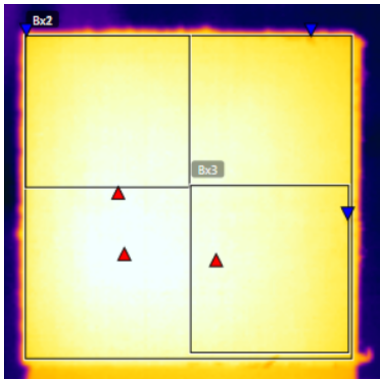


Figure 8.2: Probability density functions of the different sound areas used for the sample with three delaminated areas at a distance of 35mm from outer edge.

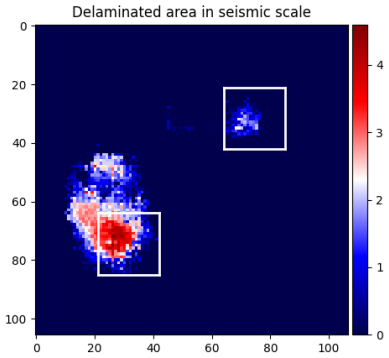
	TPR	TNR	FPR	FNR
First sound area	1.0	0.47433	0.52670	0.0
Second sound area	0.99565	0.65913	0.34087	0.00435
Unknown sound area	0.94565	0.61269	0.38731	0.05435

Table 8.2: Accuracy of the model for the sample with three delaminated areas at a distance of 35mm from outer edge.

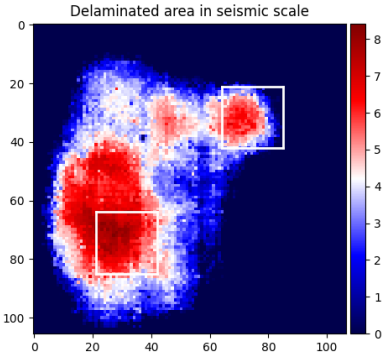
8.1.2. Specimen with a delamination of 30x30x0.1mm of paper in the bottom left corner and with a delamination of 30x30x0.5mm of pir in the upper right corner.



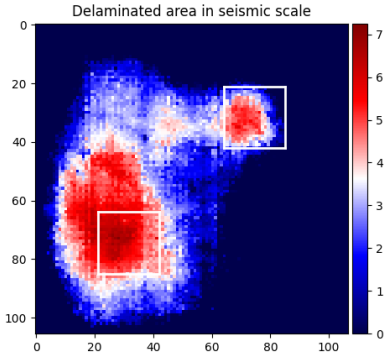
(a) Picture of the gathered data boxes from the FLIR application.



(b) Result computed with the data of the sound area of Bx2.



(c) Result computed with the the data of the sound area of Bx3.



(d) Result computed with the suggested method for a specimen without a known sound area.

Figure 8.3: Results from the specimen with a delamination of 30x30x0.1mm of paper in the bottom left corner and with a delamination of 30x30x0.5mm of pir in the upper right corner

	Average temperature [°C]	Median [°C]	Standard deviation [°C]
First sound area	46.18	46.38	0.5186
Second sound area	45.37	45.51	0.4937
Unknown sound area	44.44	44.64	0.6396

Table 8.3: Reference values of the sound areas used for the sample with delaminated areas of 0.1mm and 0.5mm thickness.

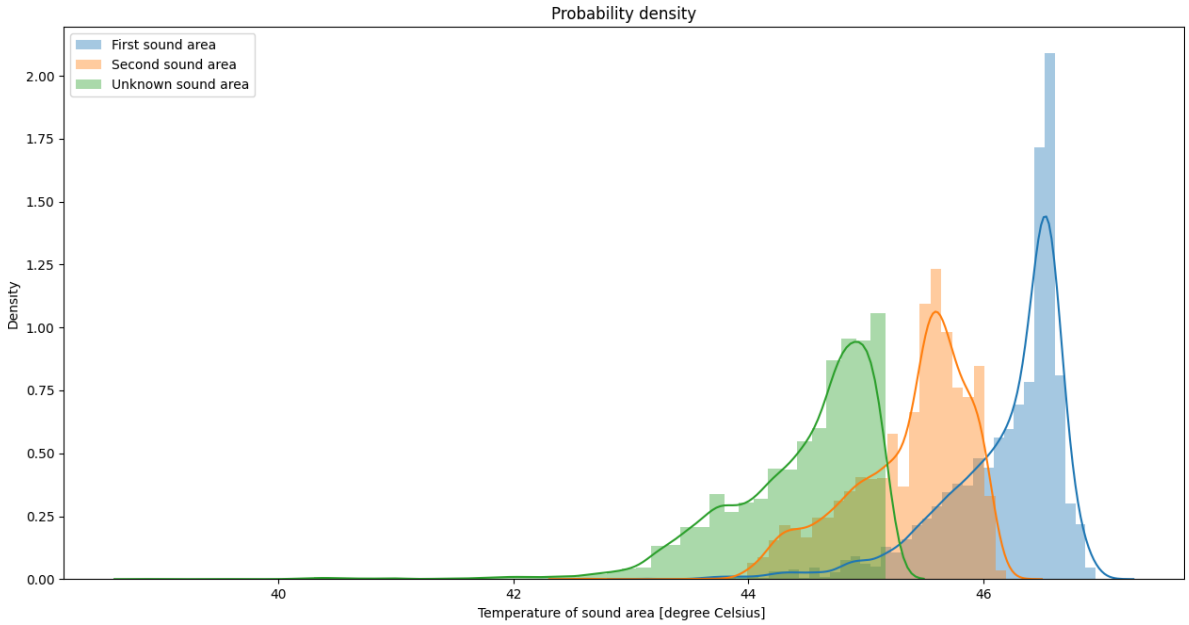
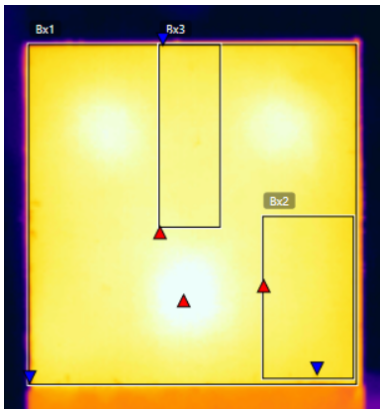


Figure 8.4: Probability density functions of the different sound areas used for the sample with delaminated areas of 0.1mm and 0.5mm thickness.

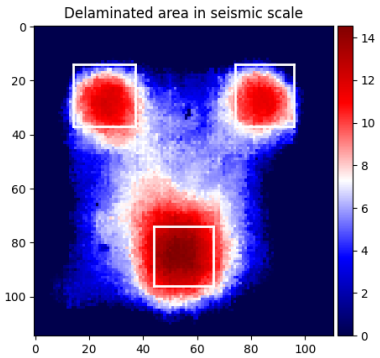
	TPR	TNR	FPR	FNR
First sound area	0.49660	0.92946	0.07054	0.50340
Second sound area	0.92177	0.56921	0.43079	0.07823
Unknown sound area	0.93197	0.57261	0.42739	0.06803

Table 8.4: Accuracy of the model for the sample with delaminated areas of 0.1mm and 0.5mm thickness.

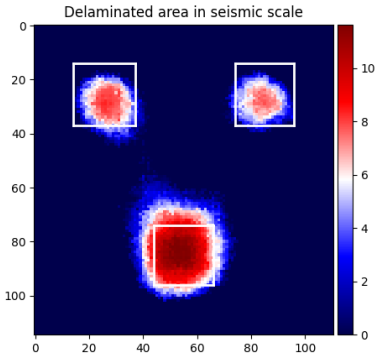
8.1.3. Specimen with three delaminations of 30x30x0.8mm of pir at a distance of 20mm ofthe outer sides.



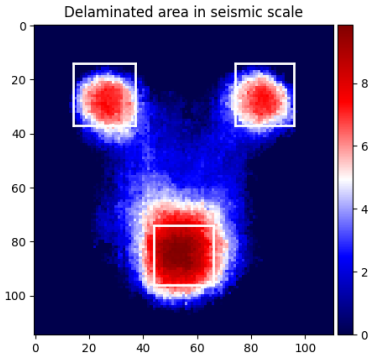
(a) Picture of the gathered data boxes from the FLIR application



(b) Result computed with the the data of the sound area of Bx2.



(c) Result computed with the the data of the sound area of Bx3.



(d) Result computed with the suggested method for a specimen without a known sound area.

Figure 8.5: Results from the specimen with three delaminations of 30x30x0.8mm of pir at a distance of 20mm of the outer sides.

	Average temperature [°C]	Median [°C]	Standard deviation [°C]
First sound area	41.10	41.17	0.5070
Second sound area	42.51	42.70	0.5316
Unknown sound area	41.51	41.71	0.8744

Table 8.5: Reference values of the sound areas used for the sample with three delaminated areas at a distance of 20mm from outer edge.

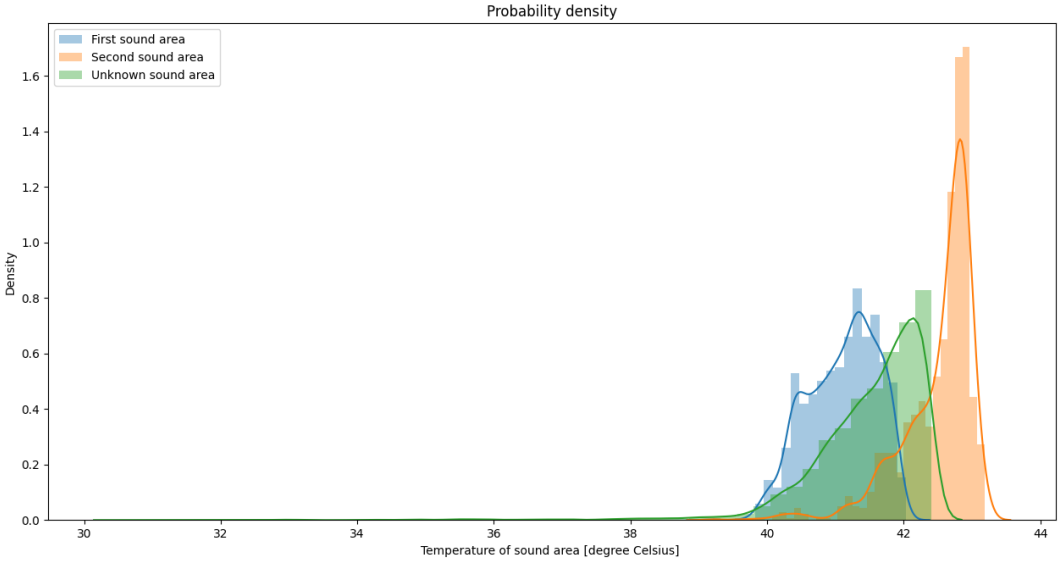


Figure 8.6: Probability density functions of the different sound areas used for the sample with three delaminated areas at a distance of 20mm from outer edge.

	TPR	TNR	FPR	FNR
First sound area	0.98785	0.46162	0.53838	0.01215
Second sound area	0.8389	0.83506	0.16494	0.1611
Unknown sound area	0.94565	0.61269	0.38731	0.05435

Table 8.6: Accuracy of the model for the sample with three delaminated areas at a distance of 20mm from outer edge.

8.2. Blind experiments

The experiment was to heat up the two specimen with a heat-flux of 1750W/m^2 for 1500 seconds and let it cool down while recording. The data from the recordings have been gathered, just one file per sample has been gathered. This file contains the temperatures per pixel of the entire sample. For both samples the results, in the form of a seismic scale picture, are presented in Figure 8.7 and Figure 8.9. The reference values of the sound areas, which is conducted by the analytical model, are presented and the probability density functions of the sound areas is also shown.

Apart from the result in the form of a seismic scale, the ratios to determine the accuracy of the model, as explained in the introduction of this chapter, are also given for both samples.

First sample

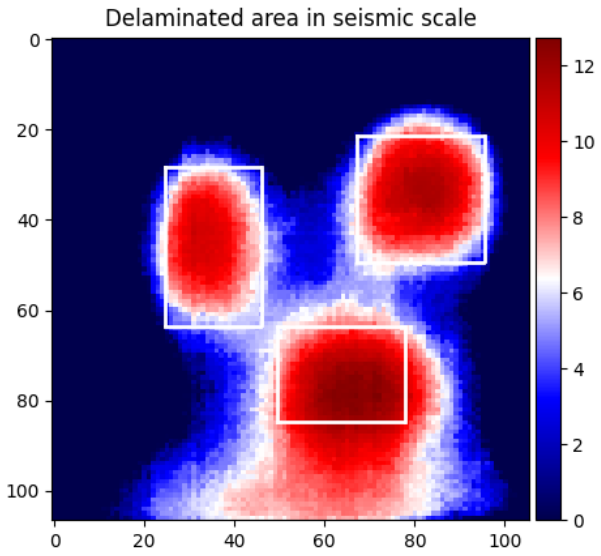


Figure 8.7: Results, presented in seismic scale, of the first sample from the blind experiments.

The reference values of the sound area of the first sample and the probability density function are as follows:

- Average temperature: 45.09 [°C]
- Median: 45.21 [°C]
- Standard deviation: 0.9518 [°C]

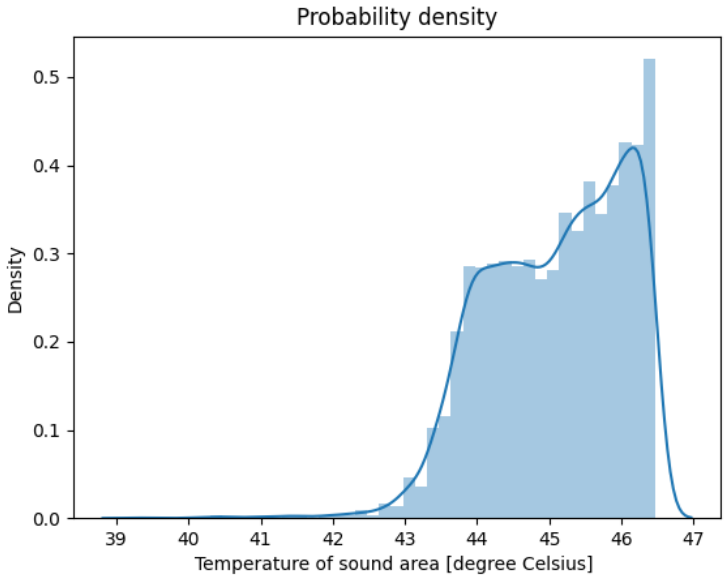


Figure 8.8: Probability density function of the sound area of the first sample from the blind experiments.

Second sample

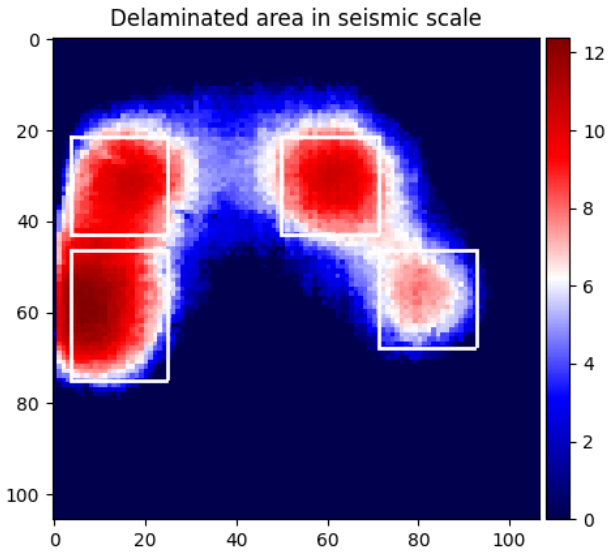


Figure 8.9: Results, presented in seismic scale, of the second sample from the blind experiments. Note: the left delamination might have shifted during the gluing procedure.

The reference values of the sound area of the second sample and the probability density function are as follows:

- Average temperature: 43.60 [°C]
- Median: 43.66 [°C]
- Standard deviation: 0.7965 [°C]

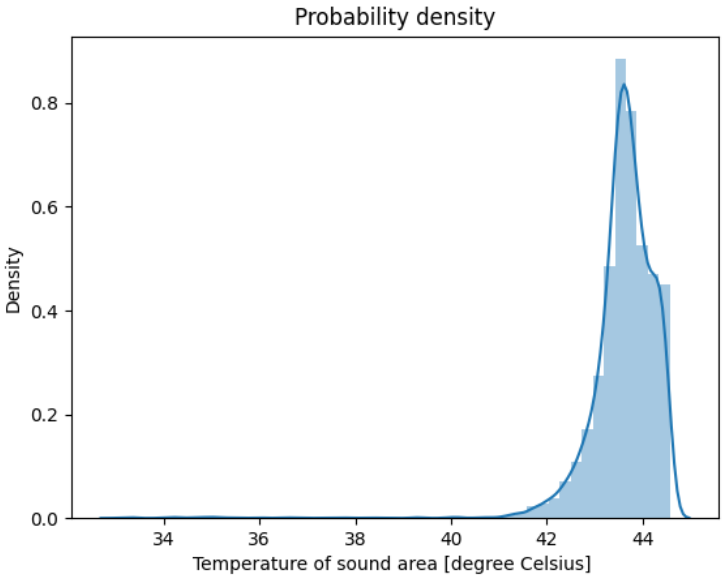


Figure 8.10: Probability density function of the sound area of the second sample from the blind experiments.

The ratios to determine the accuracy of the model are given in Table 8.7

	TPR	TNR	FPR	FNR
First sample	1.0	0.47238	0.52762	0.0
Second sample	0.9567	0.62095	0.37905	0.04135

Table 8.7: Accuracy of the model for both samples of the blind experiments.

8.3. Real experiment

In this section the results from the real experiment of the beam are presented. Two methods have been tried, the first is to use the method for an unknown sound area. This result is shown in Figure 8.11. The reference values of the sound area, produced by the analytical model, are given and the probability density function of this sound area is shown in Figure 8.12.

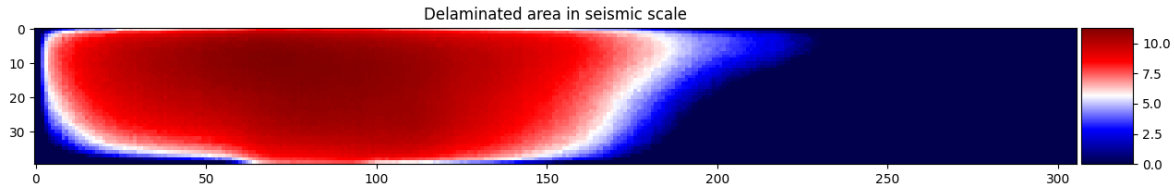


Figure 8.11: Results from beam with the method for as unknown sound area.

- Average temperature: 40.00 [°C]
- Median: 39.99 [°C]
- Standard deviation: 2.8463 [°C]

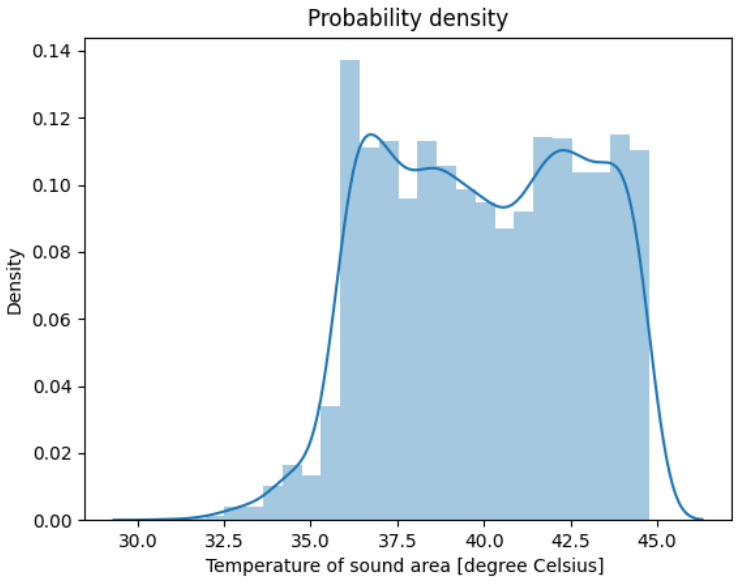


Figure 8.12: Probability density function of the sound area of the beam which is produced by the analytical model.

The other method which is used was to assign a sound area to the most right side of the beam and use this sound area in the analytical model. The location of the sound area and results from this

method are shown in Figure 8.13 and Figure 8.14 respectively. The reference values of this assigned sound area are given and the probability density function of this sound area is shown in Figure 8.15.

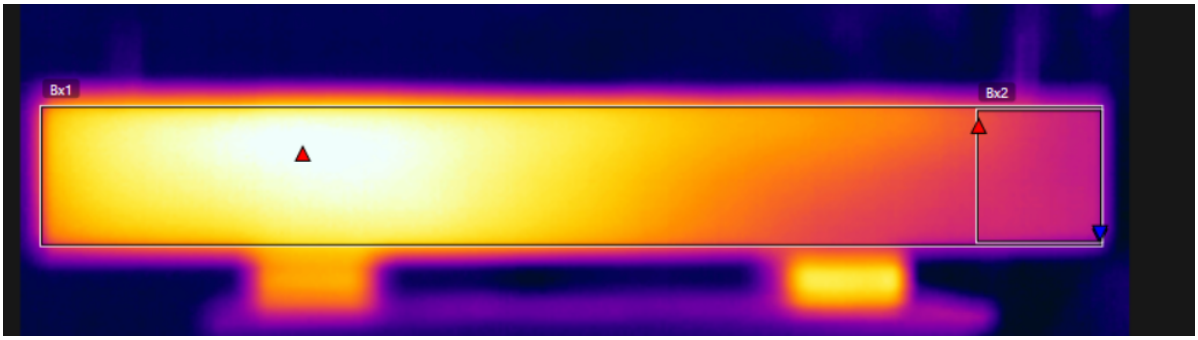


Figure 8.13: FLIR image from the beam, Bx2 represents the assigned sound area.

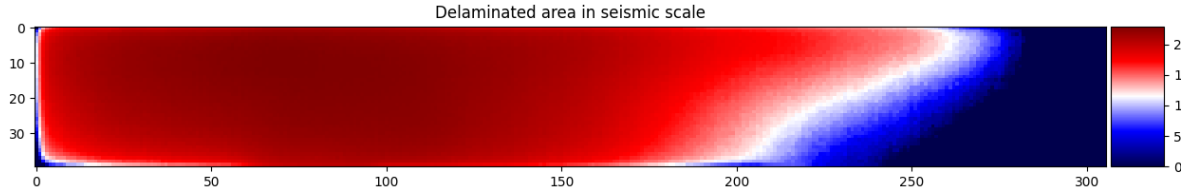


Figure 8.14: Results from beam with the method with an assigned sound area.

- Average temperature: 35.84 [°C]
- Median: 35.79 [°C]
- Standard deviation: 0.9725 [°C]

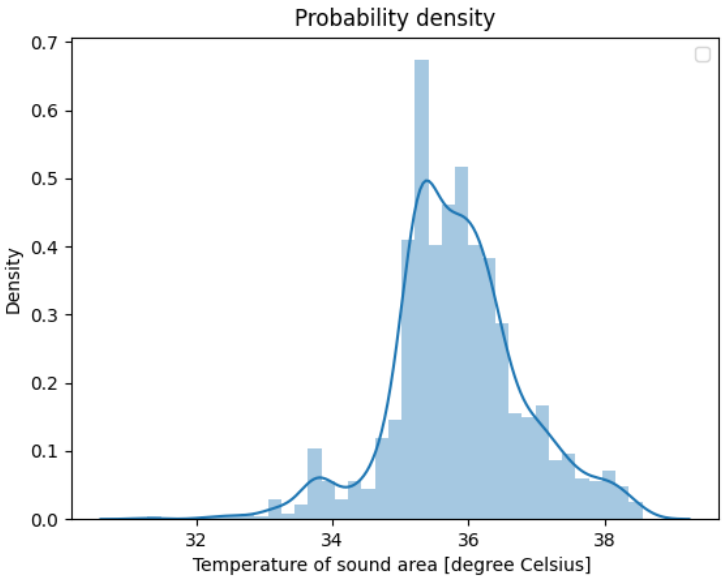


Figure 8.15: Probability density function of the sound area of the beam which is produced by the analytical model.

9 Discussion

In this chapter the results presented in chapter 8 will be discussed.

The results show that it is indeed possible to detect delaminated areas with active IRT in an objective way. It was expected to find areas with a higher temperature at the location of the delaminated areas, which the created analytical model would then pick up. As can be seen in chapter 8 this has been achieved. This study aimed to determine if active IRT would be a possible NDT method for detecting delaminations and these results are very promising.

There are a few things that need to be taken into account when analyzing the results:

- The first thing is that the accuracy of the results are quite dependent on the sound area which is chosen as a reference area. This is something that would need further research on how to optimize the choice for the reference sound area.
- The second point of attention is the fact that during the gluing process some of the PIR squares that have been used to simulate the delaminations might have shifted slighty. This is why the ratio's for the accuracy given in chapter 8 might not be 100% accurate.
- Vaseline has been used to keep the PIR squares in place during the gluing process, this might have slightly affected the thermal conductivity of the simulated delaminations. The author predicts that this might be the case with the sample where the delamination of 0.1mm is simulated with paper. This analytical model gives a delaminated area which is larger than it should be in reality, while the delaminated area of 0.5mm made of PIR insulation is picked up quite well by the analytical model.
- The thermal conductivity of the PIR squares is 0.022, whereas the thermal conductivity of air is $0.025 + 5.5d$, with d the thickness of the delamination in meters. This means that the thermal conductivity of the simulated delaminations is lower than actual delaminations. The result is that the same delamination of air is a bit harder to detect than the same delamination made of PIR. This is something to look into in follow up studies.
- The exact thermal conductivity λ of the layer of UHPFRC is unknown, this value is taken from literature study. It would be good to determine the exact thermal conductivity so that the numerical model made in COMSOL can be optimized.

The method which is suggested to be used when there is no known sound area also has one significant shortcoming which needs attention. This method is made with the assumption that there is a significant amount of sound area compared to the amount of delaminated area. The results from this method for the locations of the delaminations in the blind experiments align very well with the actual locations of the delaminations. The test with the beam clearly shows the shortcoming of this method. Almost the entire beam is delaminated and this method underestimates the amount of delamination in the interface between the NSC beam and the layer of UHPFRC. It can be seen that in this case the method of assigning a sound area gives a more accurate result than the method of not assigning a sound area. The maximum amount of delaminated volume needs to be determined in further research.

Another option would be to optimize the method for producing a sound area from the given data in another way, this could also be researched in the future.

10 Conclusion

The main goal of this research was to determine whether it is possible to detect and quantify delaminated areas, within the connection between a normal strength concrete (NSC) beam and a layer of ultra-high-performance fibre reinforced concrete (UHPFRC) of a certain thickness in an objective way, with data gathered by an infrared camera in a lab environment.

In this report a standardized approach to perform experiments with active infrared thermography (IRT) is proposed. In combination with this standardized approach, an analytical model is made to detect the location of delaminated areas in the interface between a NSC specimen and a layer of UHPFRC. The UHPFRC layer has a thickness of 10mm and is glued to the NSC specimen with an epoxy glue. The results from this standardized approach together with the analytical model are in reasonable to good agreement with the real location of delaminated areas.

Certain conclusions can be drawn for both the standardized approach as well as for the analytical model, these are presented below.

10.1. Standardized approach

For the standardized approach a heating time needs to be determined, as well as a cooling time.

The heating time is dependent on the heat-source used and the minimal delamination, which is the smallest delamination that is needed to be detected. The lower the heat-flux from the heat-source, the longer it takes to create a significant difference in temperature between the minimal delamination and a sound area.

The cooling time is mostly dependent on the environmental factors. If the standard deviation (STD) of a sound area is low, it takes less amount of time to reach the optimal cooling time. To determine the optimal cooling time the difference between the temperature above the minimal delamination and a sound area is calculated with the commercial Finite Element software COMSOL Multiphysics 5.6. The optimal cooling time is the time at which this temperature difference and the standard deviation reach the optimal ratio. This is done with the signal to noise ratio (SNR) method, which is explained a bit more below.

10.2. Analytical model

In regard to the method of detecting the delaminated area in the analytical model three methods were found in literature:

- The thresh-hold value method; a threshold value is set for the temperature difference at the surface above a sound area and a delaminated area. When the temperature difference in an area is higher than this threshold value, the area is delaminated.

- The signal to noise ratio (SNR) method; this method compares the temperature in an area to the temperature of a sound area and compares this to the STD of the temperature in a sound area. This is to take the environmental factors into account.

- The second derivative method; this method fits a 4^{th} order polynomial through the data gathered by the infrared camera. It does this for every row and column of the pixels. Each pixel has a temperature value attached to it, of the picture. Where there is an inflection point in the polynomial it is assumed that this indicates a boundary of the delaminated area.

The thresh-hold value method is difficult to make universal for different test settings, especially when the delaminated areas are unknown. For this reason, this method is not used in the current research. The SNR method and the second derivative method have been compared and the following conclusions are drawn:

- The SNR method is more accurate about the surface area of the delaminated area than the second derivative method.
- The second derivative method overestimates the surface area of the delamination.
- The SNR method is easier to interpret than the second derivative method.

This model has been used in two situations, one where a halogen lamp of 1000W was used at close distance to the specimen. This created a heat-flux of 10000[W/m^2], however, it also created a high standard deviation in the sound area due to the fact that the heat was not evenly distributed. The results from the analytical model were in this case not agreeable with the real locations of the delaminated areas. The main conclusion that can be drawn from this, is that the analytical model that was created is sensitive to uneven heat-distribution; to use this model the heat-source needs to emit an evenly distributed heat.

11

Recommendations

In this chapter, recommendations will be made regarding possible follow-up researches.

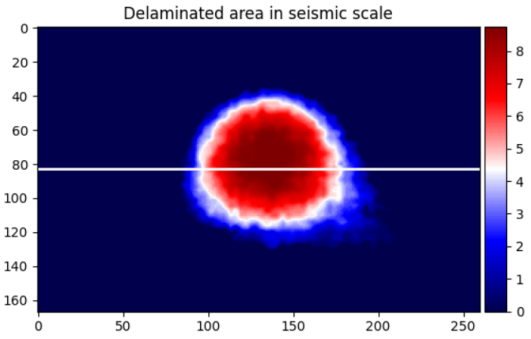
The current research shows that active IRT is a promising new NDT method for detecting delaminated areas. The analytical model which is created shows potential for implementing this method in practise. Before the analytical model can be used in practise further research needs to be done. In this chapter recommendations for further research are made.

- In the current research the smallest delaminated area which is simulated was 30x30x0.1mm and was made of paper. The results the analytical model gave were not very agreeably with reality, this might have been caused by a fault in the simulation of the delaminated area. In further research an experiment should be performed, preferably with a simulated delaminated area which has the same or roughly the same thermal conductivity as air. It is recommended, when possible, to perform this experiment with a simulated delaminated area of 30x30x0.05mm to determine whether the minimal delamination is actually detectable.
- In current research halogen lamps have been used as heating source, it was found that with halogen lamps it was possible to have a heat-flux of 1750 W/m^2. It would be valuable to research other heating methods that would be applicable in practical situation, so not only in the lab environment, but also in a practical setting. It would be of high value when this heating method would emit an evenly distributed heat with an even higher heat-flux than 1750 W/m^2. This would mean that the heating time would decrease, making the process of testing more time efficient.
- The author recommends that for further research the thermal conductivity of UHPFRC should be determined, this would increase the accuracy of the calculations of the numerical model in COMSOL.
- The method which is suggested in this report, for situations where there is no known sound area, needs to be researched further. The accuracy of this method is dependent on the relative amount of delamination in the specimen. It is recommended to research the accuracy of this method and to research what amount of delaminated area in a specimen would be the maximum amount of delaminated area, such that this method is still agreeable with reality.

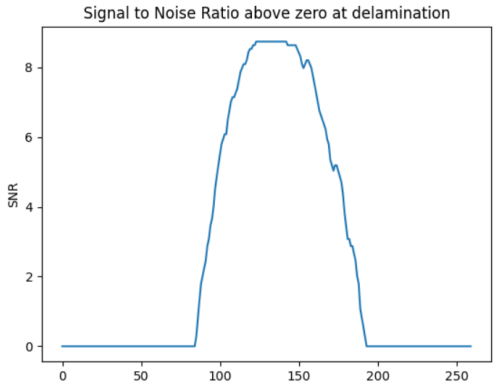
A

Results from the proof of concept experiments from grey-scale image data.

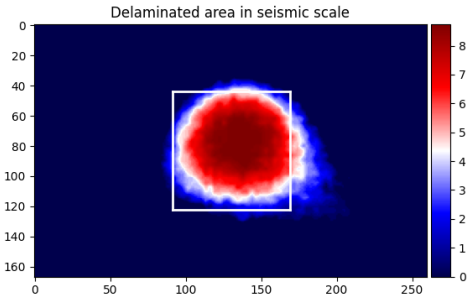
In this appendix the results from the SNR method and the second derivative method of a grey scale image is shown.



(a) Delaminated area, values of the SNR above zero shown.



(b) SNR values above zero over the mid-line of the delaminated area.



(c) Delaminated area, values of the SNR above zero shown, with a white box which indicates the known delaminated area

Figure A.1: Delamination detection with grey-scale image data, using the SNR method. Plotting the positive SNR values.

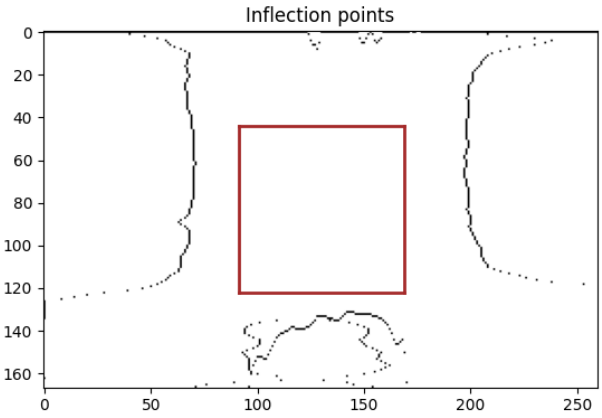


Figure A.2: Inflection points of grey-scale image from the second derivative method.

B

Heat-flux of 10000[W/m^2]

In COMSOL a simulation is done to determine the rate of increase of the RTC for different delaminated areas. Delaminated areas of the following sizes have been simulated:

- 30x30x0.1mm
- 30x30x0.5mm
- 30x30x0.8mm
- 30x30x1.0mm
- 30x30x1.3mm
- 30x30x1.5mm

For this simulation a heat-flux of 10000[W/m^2] is used, because this is the same heat-flux as is used in the first round of experiments. Because the delaminated areas in the experiments is simulated with PIR insulation, the simulation in COMSOL is done both for PIR as well as air. The temperature values have been extracted at the backside of the UHPFRC layer. This is done there in stead of at the surface of the UHPFRC layer, due to the fact that during heating the surface of the UHPFRC layer has a constant temperature over the surface due to heating directly at the surface. For these delaminated areas the times where the rate of increase of the RTC is maximum and where the rate has dropped down to 50% of the maximum rate have been conducted. These time values are found in Table B.1

Size [mm]	Pir [s]	Air [s]
30x30x0.1	18	18
30x30x0.5	25	18
30x30x0.8	25	25
30x30x1.0	25	25
30x30x1.3	26	25
30x30x1.5	26	25

(a) Time of maximum slope of RTC

Size [mm]	Pir [s]	Air [s]
30x30x0.1	69	69
30x30x0.5	100	100
30x30x0.8	100	100
30x30x1.0	100	100
30x30x1.3	100	100
30x30x1.5	100	100

(b) Time of 50% of maximum slope of RTC

Table B.1: Heating times

It can be seen that small sized delaminations are governing in the amount of minimum heating time. It is chosen to set a certain heating time that provides at least a decrease of slope up untill 50% of the maximum slope of the delaminations with an area of 30x30mm. The heating time that is chosen, taking the previous conditions into account, is 120 seconds, because two minutes is time which is easier to monitor than 100 seconds.

C

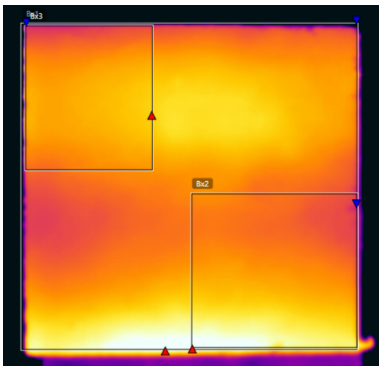
Results from close-by heating

C.0.1. Results from close-by heating with a heat-flux of 10000[W/m^2]

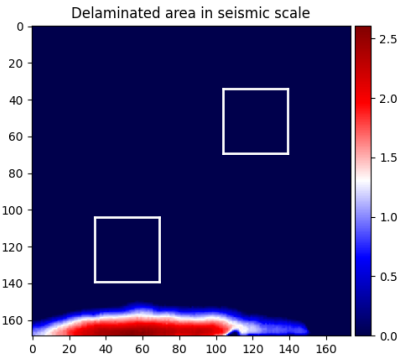
The first test setup which was discussed in section 3.2 was to heat the specimen from a close distance, which produced a heat-flux of 10000W/m^2 This was done for all the three specimen and from all three specimen data has been gathered. From the three specimen a csv file with the temperatures per pixel of every frame over the entire specimen has been extracted as well as a csv file with the temperatures per pixel of every frame over a sound area. For every specimen two sound areas have been used and the results from both are displayed as well.

As can be seen in the figures presented below, is that the results from this method are not accurate. The white boxes represent the delaminated areas which are to be detected. The delaminated areas, which are conducted with this method, do not line up with the white boxes, this is caused due to the very large difference in heating efficiency over the surface.

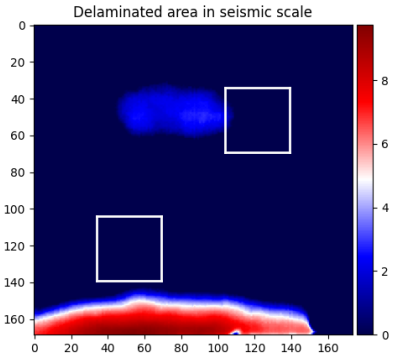
H



(a) Picture of the gathered data boxes from the FLIR application.



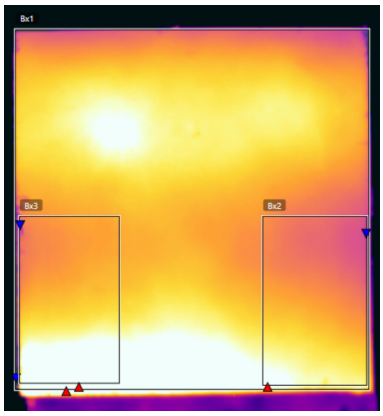
(b) Result computed with the the data of the sound area of Bx2.



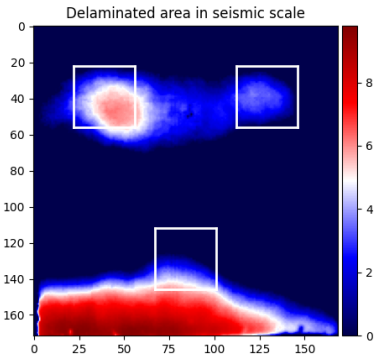
(c) Result computed with the the data of the sound area of Bx3.

Figure C.1: Results from the specimen with a delamination of 30x30x0.1mm of paper in the bottom left corner and with a delamination of 30x30x0.5mm of pir in the upper right corner

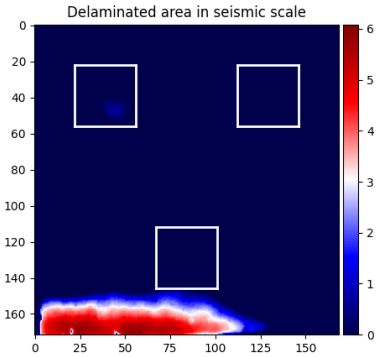
H



(a) Picture of the gathered data boxes from the FLIR application.



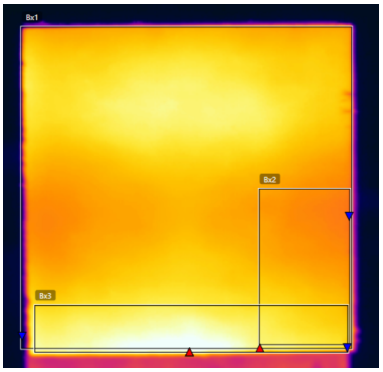
(b) Result computed with the the data of the sound area of Bx2.



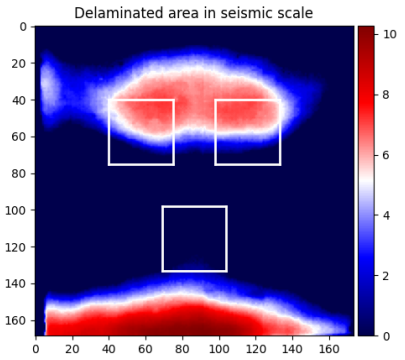
(c) Result computed with the the data of the sound area of Bx3.

Figure C.2: Results from the specimen with three delaminations of 30x30x0.8mm of pir at a distance of 20mm of the outer sides.

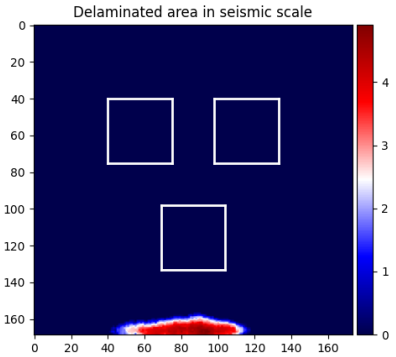
H



(a) Picture of the gathered data boxes from the FLIR application.



(b) Result computed with the the data of the sound area of Bx2.



(c) Result computed with the the data of the sound area of Bx3.

Figure C.3: Results from the specimen with three delaminations of 30x30x0.8mm of pir at a distance of 35mm of the outer sides.

D

Python scripts

In this appendix the python scripts are given in the following order:

- Second derivative method
- Calculating the heating time
- Detecting delaminations with a sound area
- Detecting delaminations without a sound area
- Scripts to which are referenced in the other scripts

D.1. Second derivative method

```python
import math
import sys
import numpy as np
import pandas
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm
import calculating_sNR as snr
from sympy.solvers import solve
from sympy.abc import x
from sympy import symbols, Eq
import cv2
import Array_to_DF as adf


def plotsecond(data):
    def Polyfit(array):
        x1 = np.arange(1, len(array) + 1, 1)
        y1 = array
        func = np.polyfit(x1, y1, 4)
        a = func[0]
        b = func[1]
        c = func[2]
        d = func[3]
        e = func[4]
        x = np.arange(1, len(array) + 1, 0.1)
        fit = a * x ** 4 + b * x ** 3 + c * x ** 2 + d * x + e

        return (a, b, c, d, e)

    def makingarray(dataline):
        ar = []
        for i in range(len(dataline)):
            ar.append(dataline[i])
        return ar

    def Secondderivative(a, b, c, d, e):
        p = np.poly1d([a, b, c, d, e])
        p2 = np.polyder(p, 2)
        return p2

    def function(x, p2):
        y = p2[0] + p2[1] * x + p2[2] * x ** 2
        return y

    y_len = len(data.iloc[:, 1])
    x_len = len(data.iloc[0, :])

    def indentionsx(data):
        points = np.zeros((len(data.iloc[:, 1]), 2))
        for i in range(len(data.iloc[:, 1])):
            dat = data.iloc[i, :]
            lenx = len(dat)

            data_true = makingarray(dat)
            x2 = np.arange(0.1, len(dat), 0.1)
            val = Polyfit(data_true)
            y1 = Secondderivative(val[0], val[1], val[2], val[3], val[4])
            x = symbols("x", positive=True)
            sol = solve(function(x, y1))
```

```
            for j in range(len(sol)):
                points[i, j] = int(sol[j])
        return points

    def indentionsy(data):
        points = np.zeros((len(data.iloc[1, :]), 2))
        for i in range(len(data.iloc[1, :])):
            dat = data.iloc[:, i]
            leny = len(dat)

            data_true = makingarray(dat)
            x2 = np.arange(0.1, len(dat), 0.1)
            val = Polyfit(data_true)
            y1 = Secondderivative(val[0], val[1], val[2], val[3], val[4])
            x = symbols("x", positive=True)
            sol = solve(function(x, y1))
            for j in range(len(sol)):
                points[i, j] = int(sol[j])
        return points

    pointsx = indentionsx(data)
    pointsy = indentionsy(data)

    print(len(pointsx[:, 0]))
    x = np.arange(0, x_len)
    y = np.arange(0, y_len)
    S = np.zeros((y_len, x_len))
    for h in y:
        print(h)
        for k in x:
            if k == pointsx[h, 0]:
                S[h, k] = 1
            elif k == pointsx[h, 1]:
                S[h, k] = 1
            elif h == pointsy[k, 0]:
                S[h, k] = 1
            elif h == pointsy[k, 1]:
                S[h, k] = 1

    pixelsize = len(S[0]) / 10
    ar = int(len(S) * 0.5)
    br = int(len(S[0]) * 0.5)
    brl = br - int(pixelsize * 3 * 0.5)
    brr = br + int(pixelsize * 3 * 0.5)
    ara = ar + int(pixelsize * 3 * 0.5)
    arb = ar - int(pixelsize * 3 * 0.5)
    x = np.arange(brl, brr + 1, 1)
    y = np.arange(arb, ara + 1, 1)
    arra = np.full(len(x), ara)
    arrb = np.full(len(x), arb)
    brrl = np.full(len(y), brl)
    brrr = np.full(len(y), brr)
    plt.plot(
        x,
        arra,
        color="brown",
        linewidth=2,
    )
    plt.plot(
        x,
```

```
        arrb,
        color="brown",
        linewidth=2,
    )
    plt.plot(
        brrl,
        y,
        color="brown",
        linewidth=2,
    )
    plt.plot(
        brrr,
        y,
        color="brown",
        linewidth=2,
    )

    plt.imshow(S, cmap="Greys")
    plt.title("Inflection points")
    plt.show()

```

D.2. Calculating the heating time

```python
import math
import sys
import numpy as np
import pandas
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm
import Average_line_comsol as alc


thickness = [0.05]  # , 0.1, 0.5, 0.8, 1, 1.3, 1.5]


pathsLPir30 = [
    r"Heating time 1750Wm2\At backside\30x30x0.05mm\Pir\10030PIRLine_data_left_0.05mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.1mm\Pir\10030PIRLine_data_left_0.1mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.5mm\Pir\10030PIRLine_data_left_0.5mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.8mm\Pir\10030PIRLine_data_left_0.8mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.0mm\Pir\10030PIRLine_data_left_1.0mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.3mm\Pir\10030PIRLine_data_left_1.3mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.5mm\Pir\10030PIRLine_data_left_1.5mm.txt",
]
pathsLAir30 = [
    r"Heating time 1750Wm2\At backside\30x30x0.05mm\Air\10030AIRLine_data_left_0.05mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.1mm\Air\10030AIRLine_data_left_0.1mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.5mm\Air\10030AIRLine_data_left_0.5mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.8mm\Air\10030AIRLine_data_left_0.8mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.0mm\Air\10030AIRLine_data_left_1.0mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.3mm\Air\10030AIRLine_data_left_1.3mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.5mm\Air\10030AIRLine_data_left_1.5mm.txt",
]
pathsRPir30 = [
    r"Heating time 1750Wm2\At backside\30x30x0.05mm\Pir\10030PIRLine_data_right_0.05mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.1mm\Pir\10030PIRLine_data_right_0.1mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.5mm\Pir\10030PIRLine_data_right_0.5mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.8mm\Pir\10030PIRLine_data_right_0.8mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.0mm\Pir\10030PIRLine_data_right_1.0mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.3mm\Pir\10030PIRLine_data_right_1.3mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.5mm\Pir\10030PIRLine_data_right_1.5mm.txt",
```

```
]
pathsRAir30 = [
    r"Heating time 1750Wm2\At backside\30x30x0.05mm\Air\10030AIRLine_data_right_0.05mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.1mm\Air\10030AIRLine_data_right_0.1mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.5mm\Air\10030AIRLine_data_right_0.5mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.8mm\Air\10030AIRLine_data_right_0.8mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.0mm\Air\10030AIRLine_data_right_1.0mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.3mm\Air\10030AIRLine_data_right_1.3mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.5mm\Air\10030AIRLine_data_right_1.5mm.txt",
]
pathsMPir30 = [
    r"Heating time 1750Wm2\At backside\30x30x0.05mm\Pir\10030PIRLine_data_mid_0.05mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.1mm\Pir\10030PIRLine_data_mid_0.1mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.5mm\Pir\10030PIRLine_data_mid_0.5mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.8mm\Pir\10030PIRLine_data_mid_0.8mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.0mm\Pir\10030PIRLine_data_mid_1.0mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.3mm\Pir\10030PIRLine_data_mid_1.3mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.5mm\Pir\10030PIRLine_data_mid_1.5mm.txt",
]
pathsMAir30 = [
    r"Heating time 1750Wm2\At backside\30x30x0.05mm\Air\10030AIRLine_data_mid_0.05mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.1mm\Air\10030AIRLine_data_mid_0.1mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.5mm\Air\10030AIRLine_data_mid_0.5mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x0.8mm\Air\10030AIRLine_data_mid_0.8mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.0mm\Air\10030AIRLine_data_mid_1.0mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.3mm\Air\10030AIRLine_data_mid_1.3mm.txt",
    #    r"Heating time 1750Wm2\At backside\30x30x1.5mm\Air\10030AIRLine_data_mid_1.5mm.txt",
]


def datasets(leftline, rightline, midline):
    data_left = pandas.read_csv(
        leftline,
        skiprows=7,
        delim_whitespace=True,
        decimal=",",
    )
    i = 0
    for i in np.arange(0, 100):
```

```
        i += 1
        if float(data_left.iloc[i + 1, 0]) - 0 == 0.0:
            break

    len_left = i + 1

    time = len(data_left) / len_left - 1

    data_right = pandas.read_csv(
        rightline,
        skiprows=7,
        delim_whitespace=True,
        decimal=",",
    )
    j = 0
    for j in np.arange(0, 100):
        j += 1
        if float(data_right.iloc[j + 1, 0]) - 0 == 0.0:
            break

    len_right = j + 1

    data_mid = pandas.read_csv(
        midline,
        skiprows=7,
        delim_whitespace=True,
        decimal=",",
    )
    k = 0
    for k in np.arange(0, 100):
        k += 1
        if float(data_mid.iloc[k + 1, 0]) - 0 == 0.0:
            break

    len_mid = k + 1

    mid1 = alc.average_line(data_mid, len_mid)[0]

    left1 = alc.average_line(data_left, len_left)[0]

    right1 = alc.average_line(data_right, len_right)[0]

    return left1, right1, mid1, time


def opthtime(c, mat, mid, left, right, time, stepsize):
    S_avg = []
    for k in range(len(left)):
        S_avg.append((left[k] + right[k]) / 2)

    def runncontrast(T_def, T_non_def):
        r = (T_def - T_non_def) / T_non_def
        return r

    mid = np.array(mid)
    S_avg = np.array(S_avg)
    runn = runncontrast(mid, S_avg)
    x1 = np.arange(0, len(runn), 1)
    x = np.arange(10, time + 1, stepsize)

```

```
    slope = []
    for i in range(len(runn) - 10):
        slope.append(runn[i + 10] - runn[i + 9])

    a = len(runn) - len(slope) + 1
    max_slope = np.max(slope)
    t_max_slope = np.argmax(slope)

    opt_slope = 0.005 * slope[np.argmax(slope)]

    diff = []
    for j in range(len(slope)):
        if j > t_max_slope:
            if slope[j] - opt_slope > 0:
                diff.append(slope[j] - opt_slope)

    Opt_time = np.argmin(diff) + a + t_max_slope

    plt.plot(x1, runn, label=f"{mat}")
    plt.title("Running Thermal contrast")
    plt.axvline(Opt_time, 0, 10, color=c, label=f"Heating time, {mat}")

    return max_slope, t_max_slope + a, Opt_time, S_avg[np.argmin(diff)]


for h in range(len(pathsRPir30)):
    leftt = datasets(pathsLPir30[h], pathsRPir30[h], pathsMPir30[h])[0]
    rightt = datasets(pathsLPir30[h], pathsRPir30[h], pathsMPir30[h])[1]
    midd = datasets(pathsLPir30[h], pathsRPir30[h], pathsMPir30[h])[2]
    time = datasets(pathsLPir30[h], pathsRPir30[h], pathsMPir30[h])[3]
    print(f"PIR 30x30 {thickness[h]}")
    print(opthtime("r", "PIR", midd, leftt, rightt, time, 1))

for h in range(len(pathsRAir30)):
    leftt = datasets(pathsLAir30[h], pathsRAir30[h], pathsMAir30[h])[0]
    rightt = datasets(pathsLAir30[h], pathsRAir30[h], pathsMAir30[h])[1]
    midd = datasets(pathsLAir30[h], pathsRAir30[h], pathsMAir30[h])[2]
    time = datasets(pathsLAir30[h], pathsRAir30[h], pathsMAir30[h])[3]
    print(f"AIR 30x30 {thickness[h]}")
    print(opthtime("purple", "Air", midd, leftt, rightt, time, 1))

plt.ylabel("RTC")

plt.xlabel("Time [s]")
plt.legend()
plt.show()
```

D.3. Detecting delaminations with a sound area

```python
import math
import sys
import numpy as np
import pandas
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm
import calculating_sNR as snr
import Old_Optimal_cooling_time as ooct
import framecalculation as fmc
import seaborn as sns
import Optimal_frame_without_knowl_del as optfn

time = 10 * 60

datafiles = [
    r"Cooling 1750Wm2\big-40cm25min_nondel.csv",
    r"Cooling 1750Wm2\big-40cm25min_nondel2.csv",
]
soundframes = []

for p in range(len(datafiles)):
    data = pandas.read_csv(
        datafiles[p],
        delimiter=";",
        skiprows=3,
        index_col=0,
        decimal=",",
        names=list(range(1000)),
    ).dropna(axis="columns", how="all")

    recording = pandas.read_csv(
        r"Cooling 1750Wm2\big-40cm25min_all.csv",
        delimiter=";",
        skiprows=3,
        index_col=0,
        decimal=",",
        names=list(range(1000)),
    ).dropna(axis="columns", how="all")

    frames_entire = fmc.amount_frames(datafiles[p])

    fps = time / frames_entire
    framl = len(data) / frames_entire

    x = int(len(data))
    a = int(x / framl)

    Framerec = []
    frame = []
    for j in range(a):
        frame.append(f"Frame{j}")
        Framerec.append(f"Frame{j}")

    runningcontrast = []

    for i in range(a):
        framla = np.arange(framl * i, framl * (i + 1))

        frame[i] = data.iloc[framla]
```

```
61
62      framl2 = len(recording) / frames_entire
63      x2 = int(len(recording))
64      a2 = int(x2 / framl2)
65      for n in range(a2):
66          framla2 = np.arange(framl2 * n, framl2 * (n + 1))
67          Framerec[n] = recording.iloc[framla2]
68
69      std = []
70      for j in range(len(frame)):
71          a = np.array(frame[j])
72          std.append(np.std(a))
73
74      dataset = [
75          r"Cooling time 1750Wm2\30x30x0.05mm\Air\10030AIRLine_data_left_0.05mm.txt",
76          r"Cooling time 1750Wm2\30x30x0.05mm\Air\10030AIRLine_data_mid_0.05mm.txt",
77          r"Cooling time 1750Wm2\30x30x0.05mm\Air\10030AIRLine_data_right_0.05mm.txt",
78      ]
79
80      datalines = ooct.datasets(dataset[0], dataset[2], dataset[1], 2)
81      left = datalines[0]
82      right = datalines[1]
83      mid = datalines[2]
84      time = datalines[3]
85
86      S_avg = []
87      for k in range(len(left)):
88          S_avg.append((left[k] + right[k]) / 2)
89
90      mid = np.array(mid)
91
92      S_avg = np.array(S_avg)
93
94      def SNR(Sd, Sa, sigmas):
95          snr = 20 * np.log10((Sd - Sa) / sigmas)
96          return snr
97
98      G = np.zeros(int(len(mid) - 1500 / 2))
99      Times = []
100     B = []
101
102     for l in np.arange(1500 / 2, len(mid) - (5 * 60) / 2, 2):
103         l = int(l)
104         t = int((((l - 1500 / 2) / time) * frames_entire))
105         print(t)
106         b = SNR(mid[l], S_avg[l], std[t])
107         B.append(b)
108
109         if b > 0:
110             Times.append(l)
111             G[l - int(1500 / 2)] = 1
112
113     framewanted = int((np.argmax(B) / len(B)) * frames_entire)
114
115     frame_ = Framerec[framewanted]
116     print(framewanted)
117     print(np.argmax(B))
118     sound_frame = frame[framewanted]
119     soundframes.append(sound_frame)
120     sns.distplot(sound_frame)
```

```
    plt.title("Probability density")
    plt.xlabel("Temperature of sound area [degree Celsius] ")
    plt.legend()
    plt.show()
    print(
        f"Maximum temp difference via COMSOL for 0.05mm delamination: {np.max(mid - S_avg)}"
    )
    print(f"Standard deviation at opt frame: {std[framewanted]}")
    print(
        f"Temp difference at opt frame via COMSOL for 0.05mm delamination: {mid[int((framewanted / frames_entire) * time) + int(1500 / 2)]- S_avg[int((framewanted / frames_entire) * time) + int(1500 / 2)]}"
    )
    avg = np.mean(np.mean(frame[framewanted]))
    std = std[framewanted]
    med = np.median(sound_frame)
    print(f"avg = {avg}, std = {std}, median = {med}")
    print(snr.SNRplot_seismicwb3(frame_, avg, std, 3.5))
    print(snr.SAMPLE3(frame_, avg, std, 3.5))


sns.distplot(soundframes[0], label=f"First sound area")
plt.title("Probability density")
plt.xlabel("Temperature of sound area [degree Celsius] ")
plt.legend()
plt.show()
```

D.4. Detecting delaminations without a sound area

```python
import math
import sys
import numpy as np
import pandas
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm
import calculating_sNR as snr
import Old_Optimal_cooling_time as ooct
import framecalculation as fmc
import seaborn as sns

time = 10 * 60


recording = pandas.read_csv(
    r"Cooling 1750Wm2\Beam\Beam_second.csv",
    delimiter=";",
    skiprows=3,
    index_col=0,
    decimal=",",
    names=list(range(500)),
).dropna(axis="columns", how="all")


frames_entire = fmc.amount_frames(r"Cooling 1750Wm2\Beam\Beam_second.csv")

fps = time / frames_entire
framl = len(recording) / frames_entire
x = int(len(recording))
a = int(x / framl)
Framerec = []
frame = []
for j in range(a):
    frame.append(f"Frame{j}")
    Framerec.append(f"Frame{j}")

runningcontrast = []

for i in range(a):
    framla = np.arange(framl * i, framl * (i + 1))

    frame[i] = recording.iloc[framla]


framl2 = len(recording) / frames_entire
x2 = int(len(recording))
a2 = int(x2 / framl2)
for n in range(a2):
    framla2 = np.arange(framl2 * n, framl2 * (n + 1))
    Framerec[n] = recording.iloc[framla2]


std = []
for j in range(len(frame)):
    a = np.array(frame[j])
    std.append(np.std(a))


dataset = [
```

```
    r"Cooling time 1750Wm2\30x30x0.05mm\Air\10030AIRLine_data_left_0.05mm.txt",
    r"Cooling time 1750Wm2\30x30x0.05mm\Air\10030AIRLine_data_mid_0.05mm.txt",
    r"Cooling time 1750Wm2\30x30x0.05mm\Air\10030AIRLine_data_right_0.05mm.txt",
]

datalines = ooct.datasets(dataset[0], dataset[2], dataset[1], 2)
left = datalines[0]
right = datalines[1]
mid = datalines[2]
time = datalines[3]

S_avg = []
for k in range(len(left)):
    S_avg.append((left[k] + right[k]) / 2)

mid = np.array(mid)


S_avg = np.array(S_avg)


def SNR(Sd, Sa, sigmas):
    snr = 20 * np.log10((Sd - Sa) / sigmas)
    return snr


G = np.zeros(int(len(mid) - 1500 / 2))
Times = []
B = []

for l in np.arange(1500 / 2, len(mid) - (5 * 60) / 2, 1):
    l = int(l)
    t = int(((((l - 1500 / 2) * 2) / time) * frames_entire))
    b = SNR(mid[l], S_avg[l], std[t])
    B.append(b)
    if b > 0:
        Times.append(l)
        G[l - int(1500 / 2)] = 1

framewanted = int((np.argmax(B) / len(B)) * frames_entire)
comsolwanted = np.argmax(B)


frame_ = Framerec[framewanted]


avg = np.mean(np.mean(frame[framewanted]))
print(f"Average first time: {avg}")
std = std[framewanted]
print(f"STD first time: {std}")
print(snr.SNRplot_seismicnb(frame_, avg, std))


tempdifcomsol = np.array(mid[int(1500 / 2) :]) - np.array(S_avg[int(1500 / 2) :])


def delamination(temp, tempdif):
    Z1 = snr.Z(temp)[0]
    y_len = snr.Z(temp)[1]
    x_len = snr.Z(temp)[2]
```

```
    avg = np.mean(np.mean(Z1))

    x = np.arange(0, x_len)
    y = np.arange(0, y_len)

    S = np.zeros((y_len, x_len))

    g = 0
    p = 0
    for h in y:
        for k in x:
            a = Z1[h][k] - avg
            v = a - 0.5 * tempdif

            p += 1
            if v > 0:
                S[h][k] = 1
                g += 1

            else:
                S[h][k] = 0

    plt.imshow(S, cmap="gray")
    plt.title("Non-delaminated area in black")
    plt.show()
    return S


def nondel(tempframe, S):
    Z1 = snr.Z(tempframe)[0]
    y_len = snr.Z(tempframe)[1]
    x_len = snr.Z(tempframe)[2]

    x = np.arange(0, x_len)
    y = np.arange(0, y_len)

    W = []
    for h in y:
        for k in x:
            if S[h][k] == 0:
                W.append(Z1[h][k])
    avgnondel = np.mean(W)
    signondel = np.std(W)
    mednondel = np.median(W)
    return avgnondel, signondel, mednondel, W


print(len(tempdifcomsol))

S1 = delamination(frame_, tempdifcomsol[comsolwanted])
nondel_avg = nondel(frame[framewanted], S1)[0]
nondel_std = nondel(frame[framewanted], S1)[1]
nondel_med = nondel(frame[framewanted], S1)[2]

print(framewanted)

print(np.argmax(B))

frame_1 = Framerec[framewanted]

```

```
avg1 = nondel_avg

std1 = nondel_std

print(f"Average second time: {avg1} ")
print(f"STD second time: {std1}")
print(snr.SNRplot_seismicunknown1(frame_1, avg1, std1))
print(snr.Unknown1(frame_1, avg1, std1))
print(f"Avergae  = {avg1}, std = {std1}, median = {nondel_med}")

sound_frame = nondel(frame[framewanted], S1)[3]


def soundplot():
    return sound_frame


sns.distplot(soundplot())
plt.title("Probability density")
plt.xlabel("Temperature of sound area [degree Celsius] ")
plt.show()
```

D.5. Scripts to which are referenced in the other scripts

```python
#%%
import math
import sys
import numpy as np
import pandas
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm
from mpl_toolkits.axes_grid1 import make_axes_locatable
from numpy import NaN, unravel_index
import plottingdelamination as pltd

# Transforming a pandas datafile into a 2d array
def Z(temp):
    y_len = len(temp.iloc[:, 0])
    x_len = len(temp.iloc[0, :])
    x = np.arange(0, x_len)
    y = np.arange(0, y_len)
    Z = np.zeros((y_len, x_len))
    for i in y:
        for j in x:
            Z[i][j] = temp.iloc[i, j]
    return (Z, y_len, x_len)


def SNRplot_seismicwb(temp, avg, sig, distance):
    Z1 = Z(temp)[0]
    y_len = Z(temp)[1]
    x_len = Z(temp)[2]

    x = np.arange(0, x_len)
    y = np.arange(0, y_len)

    def SNR(Sd, Sa, sigmas):
        snr = 20 * np.log10((Sd - Sa) / sigmas)
        return snr

    S = np.zeros((y_len, x_len))
    P = np.zeros((y_len, x_len))
    tempdifference = []
    g = 0
    p = 0
    for h in y:
        for k in x:
            tempdifference.append(Z1[h][k] - avg)
            a = SNR(Z1[h][k], avg, sig)
            P[h][k] = a
            p += 1
            if a > 0:
                S[h][k] = a
                g += 1
            else:
                S[h][k] = 0

    pixelsize = len(S[0]) / 15
    print(
        f"Actual maximum tempertature difference for actual delamination:
{np.max(tempdifference)}"
    )

```

```
    ahr = pltd.box(pixelsize, distance)[0]
    ahl = pltd.box(pixelsize, distance)[1]
    bhr = pltd.box(pixelsize, distance)[2]
    bhl = pltd.box(pixelsize, distance)[3]
    chr = pltd.box(pixelsize, distance)[4]
    chl = pltd.box(pixelsize, distance)[5]
    avu = pltd.box(pixelsize, distance)[6]
    avd = pltd.box(pixelsize, distance)[7]
    cvu = pltd.box(pixelsize, distance)[8]
    cvd = pltd.box(pixelsize, distance)[9]

    xa = np.arange(ahl, ahr + 1, 1)
    xb = np.arange(bhl, bhr + 1, 1)
    xc = np.arange(chl, chr + 1, 1)
    ya = np.arange(avu, avd + 1, 1)
    yb = np.arange(avu, avd + 1, 1)
    yc = np.arange(cvu, cvd + 1, 1)

    ya1 = np.full(len(xa), avd)
    ya2 = np.full(len(xa), avu)
    yb1 = np.full(len(xb), avd)
    yb2 = np.full(len(xb), avu)
    yc1 = np.full(len(xc), cvd)
    yc2 = np.full(len(xc), cvu)

    xa1 = np.full(len(ya), ahl)
    xa2 = np.full(len(ya), ahr)
    xb1 = np.full(len(yb), bhl)
    xb2 = np.full(len(yb), bhr)
    xc1 = np.full(len(yc), chl)
    xc2 = np.full(len(yc), chr)

    plt.plot(
        xa,
        ya1,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xa,
        ya2,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb,
        yb1,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb,
        yb2,
        color="white",
        linewidth=2,
    )

    plt.plot(
        xc,
        yc1,
```

```
        color="white",
        linewidth=2,
    )
    plt.plot(
        xc,
        yc2,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xa1,
        ya,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xa2,
        ya,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb1,
        yb,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb2,
        yb,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xc1,
        yc,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xc2,
        yc,
        color="white",
        linewidth=2,
    )

    ax = plt.gca()
    im = ax.imshow(S, cmap="seismic")
    plt.title("Delaminated area in seismic scale")
    divider = make_axes_locatable(ax)
    cax = divider.append_axes("right", size="5%", pad=0.05)
    plt.colorbar(im, cax=cax)
    plt.show()


def SNRplot_seismicunknown1(temp, avg, sig):
    Z1 = Z(temp)[0]
    y_len = Z(temp)[1]
    x_len = Z(temp)[2]

```

```
    x = np.arange(0, x_len)
    y = np.arange(0, y_len)

    def SNR(Sd, Sa, sigmas):
        snr = 20 * np.log10((Sd - Sa) / sigmas)
        return snr

    S = np.zeros((y_len, x_len))
    P = np.zeros((y_len, x_len))
    tempdifference = []
    g = 0
    p = 0
    for h in y:
        for k in x:
            tempdifference.append(Z1[h][k] - avg)
            a = SNR(Z1[h][k], avg, sig)
            P[h][k] = a
            p += 1
            if a > 0:
                S[h][k] = a
                g += 1
            else:
                S[h][k] = 0

    pixelsize = len(S[0]) / 15
    print(
        f"Actual maximum tempertature difference for actual delamination:
{np.max(tempdifference)}"
    )

    xa = np.arange(3.5 * pixelsize, 6.5 * pixelsize, 1)
    xb = np.arange(7 * pixelsize, 11 * pixelsize, 1)
    xc = np.arange(9.5 * pixelsize, 13.5 * pixelsize, 1)
    ya = np.arange(4 * pixelsize, 9 * pixelsize, 1)
    yb = np.arange(9 * pixelsize, 12 * pixelsize, 1)
    yc = np.arange(3 * pixelsize, 7 * pixelsize, 1)

    ya1 = np.full(len(xa), 4 * pixelsize)
    ya2 = np.full(len(xa), 9 * pixelsize)
    yb1 = np.full(len(xb), 9 * pixelsize)
    yb2 = np.full(len(xb), 12 * pixelsize)
    yc1 = np.full(len(xc), 3 * pixelsize)
    yc2 = np.full(len(xc), 7 * pixelsize)

    xa1 = np.full(len(ya), 3.5 * pixelsize)
    xa2 = np.full(len(ya), 6.5 * pixelsize)
    xb1 = np.full(len(yb), 7 * pixelsize)
    xb2 = np.full(len(yb), 11 * pixelsize)
    xc1 = np.full(len(yc), 9.5 * pixelsize)
    xc2 = np.full(len(yc), 13.5 * pixelsize)

    plt.plot(
        xa,
        ya1,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xa,
        ya2,
```

```
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb,
        yb1,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb,
        yb2,
        color="white",
        linewidth=2,
    )

    plt.plot(
        xc,
        yc1,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xc,
        yc2,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xa1,
        ya,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xa2,
        ya,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb1,
        yb,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb2,
        yb,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xc1,
        yc,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xc2,
```

```
        yc,
        color="white",
        linewidth=2,
    )

    ax = plt.gca()
    im = ax.imshow(S, cmap="seismic")
    plt.title("Delaminated area in seismic scale")
    divider = make_axes_locatable(ax)
    cax = divider.append_axes("right", size="5%", pad=0.05)

    plt.colorbar(im, cax=cax)
    plt.show()


def SNRplot_seismicunknown2(temp, avg, sig):
    Z1 = Z(temp)[0]
    y_len = Z(temp)[1]
    x_len = Z(temp)[2]

    x = np.arange(0, x_len)
    y = np.arange(0, y_len)

    def SNR(Sd, Sa, sigmas):
        snr = 20 * np.log10((Sd - Sa) / sigmas)
        return snr

    S = np.zeros((y_len, x_len))
    P = np.zeros((y_len, x_len))
    tempdifference = []
    g = 0
    p = 0
    for h in y:
        for k in x:
            tempdifference.append(Z1[h][k] - avg)
            a = SNR(Z1[h][k], avg, sig)
            P[h][k] = a
            p += 1
            if a > 0:
                S[h][k] = a
                g += 1
            else:
                S[h][k] = 0

    pixelsize = len(S[0]) / 15
    print(
        f"Actual maximum tempertature difference for actual delamination:
{np.max(tempdifference)}"
    )

    xa = np.arange(0.5 * pixelsize, 3.5 * pixelsize, 1)

    xb = np.arange(0.5 * pixelsize, 3.5 * pixelsize, 1)
    xc = np.arange(7 * pixelsize, 10 * pixelsize, 1)
    xd = np.arange(10 * pixelsize, 13 * pixelsize, 1)
    ya = np.arange(3 * pixelsize, 6 * pixelsize, 1)

    yb = np.arange(6.5 * pixelsize, 10.5 * pixelsize, 1)

    yc = np.arange(3 * pixelsize, 6 * pixelsize, 1)
```

```
    yd = np.arange(6.5 * pixelsize, 9.5 * pixelsize, 1)

    ya1 = np.full(len(xa), 3 * pixelsize)
    ya2 = np.full(len(xa), 6 * pixelsize)
    yb1 = np.full(len(xb), 6.5 * pixelsize)
    yb2 = np.full(len(xb), 10.5 * pixelsize)
    yc1 = np.full(len(xc), 3 * pixelsize)
    yc2 = np.full(len(xc), 6 * pixelsize)
    yd1 = np.full(len(xd), 6.5 * pixelsize)
    yd2 = np.full(len(xd), 9.5 * pixelsize)

    xa1 = np.full(len(ya), 0.5 * pixelsize)

    xa2 = np.full(len(ya), 3.5 * pixelsize)
    xb1 = np.full(len(yb), 0.5 * pixelsize)
    xb2 = np.full(len(yb), 3.5 * pixelsize)
    xc1 = np.full(len(yc), 7 * pixelsize)
    xc2 = np.full(len(yc), 10 * pixelsize)
    xd1 = np.full(len(yd), 10 * pixelsize)
    xd2 = np.full(len(yd), 13 * pixelsize)

    plt.plot(
        xa,
        ya1,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xa,
        ya2,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb,
        yb1,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb,
        yb2,
        color="white",
        linewidth=2,
    )

    plt.plot(
        xc,
        yc1,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xc,
        yc2,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xd,
```

```
418         yd1,
419         color="white",
420         linewidth=2,
421     )
422     plt.plot(
423         xd,
424         yd2,
425         color="white",
426         linewidth=2,
427     )
428 
429     plt.plot(
430         xa1,
431         ya,
432         color="white",
433         linewidth=2,
434     )
435     plt.plot(
436         xa2,
437         ya,
438         color="white",
439         linewidth=2,
440     )
441     plt.plot(
442         xb1,
443         yb,
444         color="white",
445         linewidth=2,
446     )
447     plt.plot(
448         xb2,
449         yb,
450         color="white",
451         linewidth=2,
452     )
453     plt.plot(
454         xc1,
455         yc,
456         color="white",
457         linewidth=2,
458     )
459     plt.plot(
460         xc2,
461         yc,
462         color="white",
463         linewidth=2,
464     )
465     plt.plot(
466         xd1,
467         yd,
468         color="white",
469         linewidth=2,
470     )
471     plt.plot(
472         xd2,
473         yd,
474         color="white",
475         linewidth=2,
476     )
477 
```

```
478     ax = plt.gca()
479     im = ax.imshow(S, cmap="seismic")
480     plt.title("Delaminated area in seismic scale")
481     divider = make_axes_locatable(ax)
482     cax = divider.append_axes("right", size="5%", pad=0.05)
483 
484     plt.colorbar(im, cax=cax)
485     plt.show()
486 
487 
488 def SNRplot_seismicwb2(temp, avg, sig, distance):
489     Z1 = Z(temp)[0]
490     y_len = Z(temp)[1]
491     x_len = Z(temp)[2]
492 
493     x = np.arange(0, x_len)
494     y = np.arange(0, y_len)
495 
496     def SNR(Sd, Sa, sigmas):
497         snr = 20 * np.log10((Sd - Sa) / sigmas)
498         return snr
499 
500     S = np.zeros((y_len, x_len))
501     P = np.zeros((y_len, x_len))
502     g = 0
503     p = 0
504     for h in y:
505         for k in x:
506             a = SNR(Z1[h][k], avg, sig)
507             P[h][k] = a
508             p += 1
509             if a > 0:
510                 S[h][k] = a
511                 g += 1
512             else:
513                 S[h][k] = 0
514 
515     pixelsize = len(S[0]) / 15
516 
517     ahr = pltd.twobox(pixelsize, distance)[0]
518     ahl = pltd.twobox(pixelsize, distance)[1]
519     bhr = pltd.twobox(pixelsize, distance)[2]
520     bhl = pltd.twobox(pixelsize, distance)[3]
521     avu = pltd.twobox(pixelsize, distance)[4]
522     avd = pltd.twobox(pixelsize, distance)[5]
523     bvu = pltd.twobox(pixelsize, distance)[6]
524     bvd = pltd.twobox(pixelsize, distance)[7]
525 
526     xa = np.arange(ahl, ahr + 1, 1)
527     xb = np.arange(bhl, bhr + 1, 1)
528     ya = np.arange(avu, avd + 1, 1)
529     yb = np.arange(bvu, bvd + 1, 1)
530 
531     ya1 = np.full(len(xa), avd)
532     ya2 = np.full(len(xa), avu)
533     yb1 = np.full(len(xb), bvd)
534     yb2 = np.full(len(xb), bvu)
535 
536     xa1 = np.full(len(ya), ahl)
537     xa2 = np.full(len(ya), ahr)
```

```
    xb1 = np.full(len(yb), bhl)
    xb2 = np.full(len(yb), bhr)

    plt.plot(
        xa,
        ya1,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xa,
        ya2,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb,
        yb1,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb,
        yb2,
        color="white",
        linewidth=2,
    )

    plt.plot(
        xa1,
        ya,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xa2,
        ya,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb1,
        yb,
        color="white",
        linewidth=2,
    )
    plt.plot(
        xb2,
        yb,
        color="white",
        linewidth=2,
    )

    ax = plt.gca()
    im = ax.imshow(S, cmap="seismic")
    plt.title("Delaminated area in seismic scale")
    divider = make_axes_locatable(ax)
    cax = divider.append_axes("right", size="5%", pad=0.05)
    plt.colorbar(im, cax=cax)
    plt.show()
```

```
598
599
600 def SNRplot_seismicall(SNRm, temp, avg, sig):
601     Z1 = Z(temp)[0]
602     y_len = Z(temp)[1]
603     x_len = Z(temp)[2]
604
605     x = np.arange(0, x_len)
606     y = np.arange(0, y_len)
607
608     def SNR(Sd, Sa, sigmas):
609         snr = 20 * np.log10((Sd - Sa) / sigmas)
610         return snr
611
612     P = np.zeros((y_len, x_len))
613
614     for h in y:
615         for k in x:
616             a = SNR(Z1[h][k], avg, sig)
617             P[h][k] = a
618
619     ax = plt.gca()
620     im = ax.imshow(P, cmap="seismic")
621     plt.title("Delaminated area in seismic scale")
622     divider = make_axes_locatable(ax)
623     cax = divider.append_axes("right", size="5%", pad=0.05)
624
625     plt.colorbar(im, cax=cax)
626     plt.show()
627
628
629 def SNRplot_seismic(SNRm, temp, avg, sig):
630     Z1 = Z(temp)[0]
631     y_len = Z(temp)[1]
632     x_len = Z(temp)[2]
633
634     x = np.arange(0, x_len)
635     y = np.arange(0, y_len)
636
637     def SNR(Sd, Sa, sigmas):
638         snr = 20 * np.log10((Sd - Sa) / sigmas)
639         return snr
640
641     S = np.zeros((y_len, x_len))
642     P = np.zeros((y_len, x_len))
643     g = 0
644     p = 0
645     for h in y:
646         for k in x:
647             a = SNR(Z1[h][k], avg, sig)
648             P[h][k] = a
649             p += 1
650             if a > SNRm:
651                 S[h][k] = a
652                 g += 1
653             else:
654                 S[h][k] = 0
655     ar = int(len(S) * 0.5)
656     x = np.arange(0, len(S[0]), 1)
657
```

```
    ax = plt.gca()
    im = ax.imshow(S, cmap="seismic")
    plt.title("Delaminated area in seismic scale")
    divider = make_axes_locatable(ax)
    cax = divider.append_axes("right", size="5%", pad=0.05)

    plt.colorbar(im, cax=cax)
    plt.show()


def SAMPLE2(temp, avg, sig, dist):
    Z1 = Z(temp)[0]
    y_len = Z(temp)[1]
    x_len = Z(temp)[2]
    #
    x = np.arange(0, x_len)
    y = np.arange(0, y_len)
    #
    def SNR(Sd, Sa, sigmas):
        snr = 20 * np.log10((Sd - Sa) / sigmas)
        return snr

    #
    S = np.zeros((y_len, x_len))
    g = 0
    p = 0
    for h in y:
        for k in x:
            a = SNR(Z1[h][k], avg, sig)
            p += 1
            if a > 0:
                S[h][k] = 1
                g += 1
            else:
                S[h][k] = 0
    pixelsize = len(S[0]) / 15
    TP = []
    FP = []
    TN = []
    FN = []
    for h in y:
        for k in x:
            if (
                h >= dist * pixelsize
                and k >= (15 - dist - 3) * pixelsize
                and h <= (dist + 3) * pixelsize
                and k <= (15 - dist) * pixelsize
            ):
                if S[h][k] == 1:
                    TP.append(1)
                if S[h][k] == 0:
                    FN.append(1)
            if (
                h >= (15 - dist - 3) * pixelsize
                and k >= (dist) * pixelsize
                and h <= (15 - dist) * pixelsize
                and k <= (dist + 3) * pixelsize
            ):
                if S[h][k] == 1:
                    TP.append(1)
```

```
                if S[h][k] == 0:
                    FN.append(1)

            else:
                if S[h][k] == 0:
                    TN.append(1)
                if S[h][k] == 1:
                    FP.append(1)
    TP = np.count_nonzero(TP)
    TN = np.count_nonzero(TN)
    FP = np.count_nonzero(FP)
    FN = np.count_nonzero(FN)
    TPR = TP / (TP + FN)
    TNR = TN / (TN + FP)
    FPR = FP / (FP + TN)
    FNR = FN / (FN + TP)
    return TPR, TNR, FPR, FNR


def SAMPLE3(temp, avg, sig, dist):
    Z1 = Z(temp)[0]
    y_len = Z(temp)[1]
    x_len = Z(temp)[2]
    #
    x = np.arange(0, x_len)
    y = np.arange(0, y_len)
    #
    def SNR(Sd, Sa, sigmas):
        snr = 20 * np.log10((Sd - Sa) / sigmas)
        return snr

    #
    S = np.zeros((y_len, x_len))
    g = 0
    p = 0
    for h in y:
        for k in x:
            a = SNR(Z1[h][k], avg, sig)
            p += 1
            if a > 0:
                S[h][k] = 1
                g += 1
            else:
                S[h][k] = 0
    pixelsize = len(S[0]) / 15
    TP = []
    FP = []
    TN = []
    FN = []
    for h in y:
        for k in x:
            if (
                h >= dist * pixelsize
                and k >= dist * pixelsize
                and h <= (dist + 3) * pixelsize
                and k <= (dist + 3) * pixelsize
            ):
                if S[h][k] == 1:
                    TP.append(1)
                if S[h][k] == 0:
```

```
                    FN.append(1)
            if (
                h >= dist * pixelsize
                and k >= (15 - 3 - dist) * pixelsize
                and h <= (dist + 3) * pixelsize
                and k <= (15 - dist) * pixelsize
            ):
                if S[h][k] == 1:
                    TP.append(1)
                if S[h][k] == 0:
                    FN.append(1)
            if (
                h >= (15 - 3 - dist) * pixelsize
                and k >= (15 / 2 - 1.5) * pixelsize
                and h <= (15 - dist) * pixelsize
                and k <= (15 / 2 + 1.5) * pixelsize
            ):
                if S[h][k] == 1:
                    TP.append(1)
                if S[h][k] == 0:
                    FN.append(1)
            else:
                if S[h][k] == 0:
                    TN.append(1)
                if S[h][k] == 1:
                    FP.append(1)
    TP = np.count_nonzero(TP)
    TN = np.count_nonzero(TN)
    FP = np.count_nonzero(FP)
    FN = np.count_nonzero(FN)
    TPR = TP / (TP + FN)
    TNR = TN / (TN + FP)
    FPR = FP / (FP + TN)
    FNR = FN / (FN + TP)
    return TPR, TNR, FPR, FNR


def Unknown1(temp, avg, sig):
    Z1 = Z(temp)[0]
    y_len = Z(temp)[1]
    x_len = Z(temp)[2]
    #
    x = np.arange(0, x_len)
    y = np.arange(0, y_len)
    #
    def SNR(Sd, Sa, sigmas):
        snr = 20 * np.log10((Sd - Sa) / sigmas)
        return snr

    #
    S = np.zeros((y_len, x_len))
    g = 0
    p = 0
    for h in y:
        for k in x:
            a = SNR(Z1[h][k], avg, sig)
            p += 1
            if a > 0:
                S[h][k] = 1
                g += 1
```

```
            else:
                S[h][k] = 0
    pixelsize = len(S[0]) / 15
    TP = []
    FP = []
    TN = []
    FN = []
    for h in y:
        for k in x:
            if (
                h >= 3 * pixelsize
                and k >= 9.5 * pixelsize
                and h <= 7 * pixelsize
                and k <= 13.5 * pixelsize
            ):
                if S[h][k] == 1:
                    TP.append(1)
                if S[h][k] == 0:
                    FN.append(1)
            if (
                h >= 4 * pixelsize
                and k >= 3.5 * pixelsize
                and h <= 9 * pixelsize
                and k <= 6.5 * pixelsize
            ):
                if S[h][k] == 1:
                    TP.append(1)
                if S[h][k] == 0:
                    FN.append(1)
            if (
                h >= 9 * pixelsize
                and k >= 7 * pixelsize
                and h <= 12 * pixelsize
                and k <= 11 * pixelsize
            ):
                if S[h][k] == 1:
                    TP.append(1)
                if S[h][k] == 0:
                    FN.append(1)
            else:
                if S[h][k] == 0:
                    TN.append(1)
                if S[h][k] == 1:
                    FP.append(1)
    TP = np.count_nonzero(TP)
    TN = np.count_nonzero(TN)
    FP = np.count_nonzero(FP)
    FN = np.count_nonzero(FN)
    TPR = TP / (TP + FN)
    TNR = TN / (TN + FP)
    FPR = FP / (FP + TN)
    FNR = FN / (FN + TP)
    return TPR, TNR, FPR, FNR


def Unknown2(temp, avg, sig):
    Z1 = Z(temp)[0]
    y_len = Z(temp)[1]
    x_len = Z(temp)[2]
    #
```

```
    x = np.arange(0, x_len)
    y = np.arange(0, y_len)
    #
    def SNR(Sd, Sa, sigmas):
        snr = 20 * np.log10((Sd - Sa) / sigmas)
        return snr

    #
    S = np.zeros((y_len, x_len))
    g = 0
    p = 0
    for h in y:
        for k in x:
            a = SNR(Z1[h][k], avg, sig)
            p += 1
            if a > 0:
                S[h][k] = 1
                g += 1
            else:
                S[h][k] = 0
    pixelsize = len(S[0]) / 15
    TP = []
    FP = []
    TN = []
    FN = []
    for h in y:
        for k in x:
            if (
                h >= 3 * pixelsize
                and k >= 0.5 * pixelsize
                and h <= 6 * pixelsize
                and k <= 3.5 * pixelsize
            ):
                if S[h][k] == 1:
                    TP.append(1)
                if S[h][k] == 0:
                    FN.append(1)
            if (
                h >= 6.5 * pixelsize
                and k >= 0.5 * pixelsize
                and h <= 10.5 * pixelsize
                and k <= 4.5 * pixelsize
            ):
                if S[h][k] == 1:
                    TP.append(1)
                if S[h][k] == 0:
                    FN.append(1)
            if (
                h >= 3 * pixelsize
                and k >= 7 * pixelsize
                and h <= 6 * pixelsize
                and k <= 10 * pixelsize
            ):
                if S[h][k] == 1:
                    TP.append(1)
                if S[h][k] == 0:
                    FN.append(1)
            if (
                h >= 6.5 * pixelsize
                and k >= 10 * pixelsize
```

```
                and h <= 9.5 * pixelsize
                and k <= 13 * pixelsize
            ):
                if S[h][k] == 1:
                    TP.append(1)
                if S[h][k] == 0:
                    FN.append(1)
            else:
                if S[h][k] == 0:
                    TN.append(1)
                if S[h][k] == 1:
                    FP.append(1)
    TP = np.count_nonzero(TP)
    TN = np.count_nonzero(TN)
    FP = np.count_nonzero(FP)
    FN = np.count_nonzero(FN)
    TPR = TP / (TP + FN)
    TNR = TN / (TN + FP)
    FPR = FP / (FP + TN)
    FNR = FN / (FN + TP)
    return TPR, TNR, FPR, FNR


def SNRplot_seismicwb3(temp, avg, sig):
    Z1 = Z(temp)[0]
    y_len = Z(temp)[1]
    x_len = Z(temp)[2]

    x = np.arange(0, x_len)
    y = np.arange(0, y_len)

    def SNR(Sd, Sa, sigmas):
        snr = 20 * np.log10((Sd - Sa) / sigmas)
        return snr

    S = np.zeros((y_len, x_len))
    P = np.zeros((y_len, x_len))
    g = 0
    p = 0
    for h in y:
        for k in x:
            a = SNR(Z1[h][k], avg, sig)
            P[h][k] = a
            p += 1
            if a > 0:
                S[h][k] = a
                g += 1
            else:
                S[h][k] = 0

    pixelsize = len(S[0]) / 10
    ar = int(len(S) * 0.5)
    br = int(len(S[0]) * 0.5)
    brl = br - int(pixelsize * 3 * 0.5)
    brr = br + int(pixelsize * 3 * 0.5)
    ara = ar + int(pixelsize * 3 * 0.5)
    arb = ar - int(pixelsize * 3 * 0.5)
    x = np.arange(brl, brr + 1, 1)
    y = np.arange(arb, ara + 1, 1)
    arra = np.full(len(x), ara)
```

```
    arrb = np.full(len(x), arb)
    brrl = np.full(len(y), brl)
    brrr = np.full(len(y), brr)
    plt.plot(
        x,
        arra,
        color="white",
        linewidth=2,
    )
    plt.plot(
        x,
        arrb,
        color="white",
        linewidth=2,
    )
    plt.plot(
        brrl,
        y,
        color="white",
        linewidth=2,
    )
    plt.plot(
        brrr,
        y,
        color="white",
        linewidth=2,
    )

    ax = plt.gca()
    im = ax.imshow(S, cmap="seismic")
    plt.title("Delaminated area in seismic scale")
    divider = make_axes_locatable(ax)
    cax = divider.append_axes("right", size="5%", pad=0.05)

    plt.colorbar(im, cax=cax)
    plt.show()


def SNRplot_seismicnb(temp, avg, sig):
    Z1 = Z(temp)[0]
    y_len = Z(temp)[1]
    x_len = Z(temp)[2]

    x = np.arange(0, x_len)
    y = np.arange(0, y_len)

    def SNR(Sd, Sa, sigmas):
        snr = 20 * np.log10((Sd - Sa) / sigmas)
        return snr

    S = np.zeros((y_len, x_len))
    P = np.zeros((y_len, x_len))
    g = 0
    p = 0
    for h in y:
        for k in x:
            a = SNR(Z1[h][k], avg, sig)
            P[h][k] = a
            p += 1
            if a > 0:
```

```
                S[h][k] = a
                g += 1
            else:
                S[h][k] = 0

    ar1 = np.full(len(S[0]), len(S) * 0.25)
    ar2 = np.full(len(S[0]), len(S) * 0.5)
    ar3 = np.full(len(S[0]), len(S) * 0.75)

    x1 = np.arange(0, len(S[0]), 1)
    plt.plot(x1, ar1, color="white")
    plt.plot(x1, ar2, color="white")
    plt.plot(x1, ar3, color="white")
    ax = plt.gca()
    im = ax.imshow(S, cmap="seismic")
    plt.title("Delaminated area in seismic scale")
    divider = make_axes_locatable(ax)
    cax = divider.append_axes("right", size="5%", pad=0.05)

    plt.colorbar(im, cax=cax)
    plt.show()
    plt.plot(x1, S[int(len(S) * 0.25)])
    plt.title("SNR ratio above most upper line over the beam")
    plt.legend()
    plt.show()
    plt.plot(x1, S[int(len(S) * 0.5)])
    plt.title("SNR ratio above most middle line over the beam")
    plt.show()
    plt.plot(x1, S[int(len(S) * 0.75)])
    plt.title("SNR ratio above most lower line over the beam")
    plt.show()

```

```python
import math
import sys
import numpy as np
import pandas
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm
import csv


def average_line(data, lent):
    data = data.iloc[:, 1]

    x = int(len(data))
    a = int(x / lent)
    arrays = []
    for j in range(a):
        arrays.append(f"array{j}")

    for i in range(a):
        arr = []
        for j in range(lent * i, lent * (i + 1)):
            arr.append(float(data[j]))
        arrays[i] = arr

    average = []
    for h in range(len(arrays)):
        av = np.mean(arrays[h])
        average.append(av)

    return average, len(average)


def datasets(leftline, rightline, midline, stepsize):
    data_left = pandas.read_csv(
        leftline,
        skiprows=7,
        delim_whitespace=True,
        decimal=",",
    )
    i = 0
    for i in np.arange(0, 100):
        i += 1
        if float(data_left.iloc[i + 1, 0]) - 0 == 0.0:
            break

    len_left = i + 1

    time = len(data_left) / len_left - 1

    data_right = pandas.read_csv(
        rightline,
        skiprows=7,
        delim_whitespace=True,
        decimal=",",
    )
    j = 0
    for j in np.arange(0, 100):
        j += 1
        if float(data_right.iloc[j + 1, 0]) - 0 == 0.0:
```

```
            break

    len_right = j + 1

    data_mid = pandas.read_csv(
        midline,
        skiprows=7,
        delim_whitespace=True,
        decimal=",",
    )
    k = 0
    for k in np.arange(0, 100):
        k += 1
        if float(data_mid.iloc[k + 1, 0]) - 0 == 0.0:
            break

    len_mid = k + 1

    mid1 = average_line(data_mid, len_mid)[0]

    left1 = average_line(data_left, len_left)[0]

    right1 = average_line(data_right, len_right)[0]

    return left1, right1, mid1, time * stepsize


def optctime(mid, left, right, time, stepsize):
    S_avg = []
    for k in range(len(left)):
        S_avg.append((left[k] + right[k]) / 2)

    mid = np.array(mid)

    S_avg = np.array(S_avg)

    def runncontrast(T_def, T_non_def):
        r = (T_def - T_non_def) / T_non_def
        return r

    runn = runncontrast(mid, S_avg)

    x = np.arange(0, time + stepsize, stepsize)
    arg = int(np.argmax(runn))

    h = arg * stepsize + 1
    tempdif = mid[arg] - S_avg[arg]
    plt.plot(x, runn, label="RTC")
    plt.axvline(h, 0, 10, color="r", label="Max. RTC")
    plt.title("Running Thermal Contrast over time")
    plt.xlabel("time [s]")
    plt.ylabel("running thermal contrast")
    plt.legend()
    plt.show()
    return h, tempdif, np.max(runn), S_avg[arg]

```

```python
import math
import sys
import numpy as np
import pandas
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm


def amount_frames(datapath):

    data = pandas.read_csv(
        datapath,
        skiprows=3,
        delimiter=";",
        names=list(range(500)),
    ).dropna(axis="columns", how="all")

    a = np.array(data.iloc[:, 0])

    isnandata = pandas.isna(a)

    frames = len(a) - np.count_nonzero(isnandata)
    return frames

```

Bibliography

[1] M. H. S. van H.-M. Geesteranus. Kamerstuk 34550-XII, nr. 60 | Overheid.nl > Officiële bekendmakingen, 11 2016. URL https://zoek.officielebekendmakingen.nl/kst-34550-XII-60.html.

[2] M. H. S. van H.-M. Geesteranus. Vaststelling van de begrotingsstaten van het Ministerie van Infrastructuur en Milieu (XII) voor het jaar 2017. Technical report, 2017. URL https://www.eerstekamer.nl/behandeling/20170710/brief_regering_onderhoud/document3/f=/vkg1d0wfiszw.pdf.

[3] JC Gu, S Unjoh, and H Naito. Detectability of delamination regions using infrared thermography in concrete members strengthened by CFRP jacketing. *Composite Structures*, 245:112328, 2020. doi: 10.1016/j.compstruct.2020.112328.

[4] L Guillaumat, J Batsale, and D Mourand. Real time infra-red image processing for the detection of delamination in composite plates. *Composites Part A: Applied Science and Manufacturing*, 35 (7-8):939–944, 2004. doi: 10.1016/j.compositesa.2004.01.021.

[5] S. Hiasa, University of Central Florida. College of Engineering and Computer Science", Environmental University of Central Florida. Department of Civil, and Construction Engineering. *Investigation of Infrared Thermography for Subsurface Damage Detection of Concrete Structures*. Amsterdam University Press, Amsterdam, Netherlands, 2016.

[6] S Huang, L Li, H Yang, and K Shi. NDE of composites delamination by infrared thermography. *Nondestructive Evaluation and Health Monitoring of Aerospace Materials and Composites II*, pages 219–223, 2003. doi: 10.1117/12.484090.

[7] Jungwon Huh, Van Mac, Quang Tran, Ki-Yeol Lee, Jong-In Lee, and Choonghyun Kang. Detectability of Delamination in Concrete Structure Using Active Infrared Thermography in Terms of Signal-to-Noise Ratio. *Applied Sciences*, 8(10):1986, 2018. doi: 10.3390/app8101986.

[8] T. James. Thermal Properties of Paper, 03 2019. URL https://sciencing.com/thermal-properties-paper-6893512.html.

[9] L Junyan, L Liqiang, and W Yang. Experimental study on active infrared thermography as a NDI tool for carbon–carbon composites. *Composites Part B: Engineering*, 45(1):138–147, 2013. doi: 10.1016/j.compositesb.2012.09.006.

[10] F Khan and I Bartoli. Detection of delamination in concrete slabs combining infrared thermography and impact echo techniques: a comparative experimental study. *Structural Health Monitoring and Inspection of Advanced Materials, Aerospace, and Civil Infrastructure 2015*, pages 1–12, 2015. doi: 10.1117/12.2084096.

[11] Kingspan, n.d.

[12] E Kuhn, E Valot, and P Herve. A comparison between thermosonics and thermography for delamination detection in polymer matrix laminates. *Composite Structures*, 94(3):1155–1164, 2012. doi: 10.1016/j.compstruct.2011.10.008.

[13] W.L. Lai, S.C. Kou, C.S. Poon, W.F. Tsang, and C.C. Lai. Characterization of the deterioration of externally bonded CFRP-concrete composites using quantitative infrared thermography. *Cement and Concrete Composites*, 32(9):740–746, 2010. doi: 10.1016/j.cemconcomp.2010.03.008.

[14] CE Metz. Receiver Operating Characteristic Analysis: A Tool for the Quantitative Evaluation of Observer Performance and Imaging Systems. *Journal of the American College of Radiology*, 3 (6):413–422, 2006. doi: 10.1016/j.jacr.2006.02.021.

[15] B Nagy, SG Nehme, and D Szagri. Thermal Properties and Modeling of Fiber Reinforced Concretes. *Energy Procedia*, 78:2742–2747, 2015. doi: 10.1016/j.egypro.2015.11.616.

[16] Tarek Omar, Moncef L. Nehdi, and Tarek Zayed. Infrared thermography model for automated detection of delamination in RC bridge decks. *Construction and Building Materials*, 168:313–327, 2018. doi: 10.1016/j.conbuildmat.2018.02.126.

[17] A. Sinha, O.S. Sastry, and R. Gupta. Detection and characterisation of delamination in PV modules by active infrared thermography. *Nondestructive Testing and Evaluation*, 31(1):1–16, 2015. doi: 10.1080/10589759.2015.1034717.

[18] TU Delft. *Lecture notes CTBB4220*. TU Delft, 2000.

[19] Q Wang, Q Hu, J Qiu, C Pei, X Li, and H Zhou. Using differential spread laser infrared thermography to detect delamination and impact damage in CFRP. *Infrared Physics Technology*, 106: 103282, 2020. doi: 10.1016/j.infrared.2020.103282.